

# A study of Recombination Operators for the Cyclic Bandwidth Problem

Jintong Ren<sup>1</sup>, Jin-Kao Hao<sup>1,2\*</sup>, Eduardo Rodriguez-Tello<sup>3</sup>

<sup>1</sup> LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49095 Angers, France

<sup>2</sup> Institut Universitaire de France, 1 rue Descartes, 75231 Paris, France

<sup>3</sup> 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  $\varphi$  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

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

17 As a well-known meta-heuristic framework [12,17], memetic algorithms (MAs)  
 18 have been widely used to solve a large number of NP-hard problems [5, 11, 13,  
 19 28, 29]. For permutation problems, MAs have also reported good performances  
 20 for the Traveling Salesman Problem (TSP) [8, 16], the Quadratic Assignment  
 21 Problem [2], and other bandwidth problems [1, 20].

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

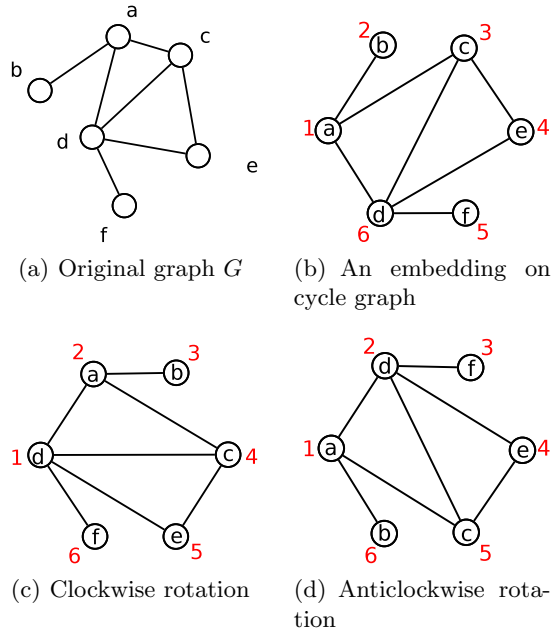
## 38 2 Memetic Algorithm for CBP

### 39 2.1 Search Space, Representation, Fitness Function

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

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

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



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

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

## 65 2.2 General procedure

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

75 improved solution is used to update the population (Section 2.6). This process  
76 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  $\varphi^*$ 
3:  $P = \{\varphi^1, \varphi^2, \dots, \varphi^{|P|}\} \leftarrow Init\_Population()$ 
4:  $\varphi^* \leftarrow Best(P)$ 
5: for  $i = 1$  to  $|P|$  do
6:    $\varphi^i \leftarrow Local\_Search(\varphi^i)$ 
7:   if  $B_C(G, \varphi^i) < B_C(G, \varphi^*)$  then
8:      $\varphi^* \leftarrow \varphi^i$ 
9:   end if
10: end for
11: for  $j = 1$  to  $MaxGene$  do
12:    $\varphi_F, \varphi_M \leftarrow Parent\_Selection(P)$ 
13:    $\varphi_C \leftarrow Recombination\_Sol(\varphi_F, \varphi_M)$ 
14:    $\varphi_C \leftarrow Local\_Search(\varphi_C)$ 
15:   if  $B_C(G, \varphi_C) < B_C(G, \varphi^*)$  then
16:      $\varphi^* \leftarrow \varphi_C$ 
17:   end if
18:    $P \leftarrow Update\_Pop(\varphi_C, P)$ 
19:    $j \leftarrow j + 1$ 
20: end for
21: return  $\varphi^*$ 

```

---

### 77 2.3 Initialization

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

### 83 2.4 Local search

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

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

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

92 where  $A(u)$  denotes the set of vertices adjacent to  $u$  of cardinality  $\text{deg}(u)$ . Then  
 93 the set of critical vertices is given by:

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

94 The neighborhood is defined as follows:

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

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

## 103 2.5 Recombination

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

## 108 2.6 Updating population

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

## 113 3 Recombination operators

114 There are several recombination operators that are already applied to permu-  
 115 tation problems [6, 8, 9, 18, 26]. We consider five crossover operators introduced  
 116 below. It is worth noting that all the recombination operations work with the  $\gamma$   
 117 representation mentioned in Section 2.1.

### 118 3.1 Order Crossover

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

### 133 3.2 Order-based Crossover

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

### 144 3.3 Cycle Crossover

145 The Cycle Crossover operator (CX) [18] seeks a way to preserve the common  
 146 information in both parent solutions. Two new offspring solutions  $\varphi_{C1}$  and  $\varphi_{C2}$   
 147 are created from two parents  $\varphi_F$  and  $\varphi_M$  where the number of each position  
 148 in  $\varphi_{C1}$  and  $\varphi_{C2}$  is decided by the number of the corresponding position of one  
 149 parent. For example, we consider two parent solutions  $\varphi_F=(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$  and  
 150  $\varphi_M=(2\ 4\ 6\ 8\ 7\ 5\ 3\ 1)$ . Firstly, the number of the first position of  $\varphi_{C1}$  could be  
 151 1 or 2, Supposing that we pick 1 here  $(1\ + + + + + + +)$ . Then, the number  
 152 of the eighth position could not be 1 because it is already assigned to the first  
 153 position, hence we allocate it with a number from  $\varphi_F$  to get  $(1\ + + + + + 8)$ .  
 154 After that, we find the position of  $\varphi_M$  whose number is 8 and assign the number  
 155 of  $\varphi_F$  to the corresponding position of  $\varphi_{C1}$ . We repeat the same operation and  
 156 find that the fourth and the second number of  $\varphi_{C1}$  come from  $\varphi_F$ , which leads to  
 157  $(1\ 2\ + 4\ + + + 8)$ . For the rest of the positions, we fill them with the numbers  
 158 from  $\varphi_M$  to obtain a complete offspring solution  $\varphi_{C1}=(1\ 2\ 6\ 4\ 7\ 5\ 3\ 8)$ . Similarly,  
 159 we could get the other offspring solution  $\varphi_{C2}=(2\ 4\ 3\ 8\ 5\ 6\ 7\ 1)$ .

### 160 3.4 Partially Mapped Crossover

161 The Partially Mapped Crossover operator (PMX) [9] passes the absolute position  
 162 information from the parent solutions to the offspring solutions. An offspring  
 163 solution gets a substring from one parent and its remaining positions take the  
 164 values of the other parent. For example, we consider again  $\varphi_F=(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$   
 165 and  $\varphi_M=(2\ 4\ 6\ 8\ 7\ 5\ 3\ 1)$ . At the beginning, two random cut points are chosen  
 166 for both parent solutions:  $\varphi_F=(1\ 2\ 3\ | 4\ 5\ 6\ | 7\ 8)$  and  $\varphi_M=(2\ 4\ 6\ | 8\ 7\ 5\ | 3$   
 167  $1)$ . Then we pass the information between the two cut points to the offspring  
 168 solutions:  $\varphi_{C1}=(+ + +\ | 4\ 5\ 6\ | + +)$  and  $\varphi_{C2}=(+ + +\ | 8\ 7\ 5\ | + +)$ . Also,  
 169 we get the mapping for the substrings between the two cut points:  $4\leftrightarrow 8$ ,  $5\leftrightarrow 7$ ,  
 170  $6\leftrightarrow 5$ . After that, the other positions of the offspring solutions are filled with the  
 171 other parent solution, hence we get  $\varphi_{C1}=(2\ 4\ 6\ | 4\ 5\ 6\ | 3\ 1)$  and  $\varphi_{C2}=(1\ 2\ 3$   
 172  $| 8\ 7\ 5\ | 7\ 8)$ . For the duplicate labels in the solution, we use the mapping of  
 173 substrings to replace the repeated numbers. In this case,  $5\leftrightarrow 7$  and  $6\leftrightarrow 5$  result in  
 174  $6\leftrightarrow 7$ . Therefore, the offspring solutions are  $\varphi_{C1}=(2\ 8\ 7\ | 4\ 5\ 6\ | 3\ 1)$  and  $\varphi_{C2}=(1$   
 175  $2\ 3\ | 8\ 7\ 5\ | 6\ 4)$ .

### 176 3.5 Distance Preserved Crossover

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

## 187 4 Experimental results

### 188 4.1 Instances and settings

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

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

## 202 4.2 Computational results

203 Table 1 outlines the computational results of our MA variants with the 5 differ-  
 204 ent recombination operators. The columns “Best” and “Avg” list the best and  
 205 average objective values found. According to the definition introduced in Section  
 206 1, a smaller objective value indicates a better result. Table 1 shows that the al-  
 207 gorithm with OX2 obtains the best results not only in terms of “Best” but also  
 208 in terms of “Avg” over the 20 test instances. From the average values listed in  
 209 the last row, we find that OX2 is a much more suitable operator than the other  
 210 operators for CBP. Also, the non-parametric Friedman test on the 5 groups of  
 211 best results leads to a  $p$ -value= $6.71e-14 < 0.05$ , confirming that there exists a  
 statistically significant difference among the compared results.

Graph	CX		DPX		OX		OX2		PMX	
	Best	Avg	Best	Avg	Best	Avg	Best	Avg	Best	Avg
nos6	327	331.28	327	329.74	266	287.98	<b>216</b>	<b>227.84</b>	327	331.98
path1000	461	475.42	462	474.02	254	301.04	<b>226</b>	<b>247.54</b>	468	482.68
nos4	44	46.12	43	45.24	32	39.32	<b>28</b>	<b>34.48</b>	42	45.78
tree10x2	39	42.72	35	40.72	<b>28</b>	32.50	<b>28</b>	<b>29.26</b>	36	41.56
cycle1000	457	476.66	466	473.38	252	296.98	<b>226</b>	<b>246.94</b>	459	480.86
mesh2D8x25	88	93.04	89	91.82	59	75.18	<b>57</b>	<b>62.94</b>	87	93.28
caterpillar29	203	211.48	203	208.70	138	162.98	<b>100</b>	<b>127.32</b>	198	210.14
mesh3D6	102	103.88	101	102.96	86	93.08	<b>73</b>	<b>78.26</b>	102	104.28
hypercube11	1022	1022.76	1022	1022.14	1019	1021.26	<b>952</b>	<b>1010.48</b>	1022	1022.54
cycle475	200	215.16	206	213.36	105	128.36	<b>99</b>	<b>110.76</b>	192	217.30
mesh2D28x30	409	413.40	410	412.06	336	371.76	<b>270</b>	<b>287.46</b>	406	414.06
mesh3D11	660	662.04	660	661.28	625	650.30	<b>507</b>	<b>522.82</b>	660	662.40
can_715	354	355.80	355	355.14	347	353.92	<b>293</b>	<b>316.70</b>	354	355.74
impcol_b	28	28.46	27	27.96	25	27.22	<b>20</b>	<b>26.72</b>	28	28.00
path475	202	214.50	206	212.86	112	132.24	<b>102</b>	<b>112.94</b>	189	217.56
494_bus	220	230.76	222	228.72	135	165.74	<b>128</b>	<b>138.62</b>	216	233.38
tree21x2	199	212.08	203	208.96	139	171.34	<b>124</b>	<b>140.84</b>	200	210.68
caterpillar44	481	493.28	479	491.24	340	400.78	<b>281</b>	<b>321.70</b>	480	495.60
impcol_d	207	209.60	207	208.80	190	202.98	<b>159</b>	<b>169.74</b>	208	209.80
tree2x9	475	489.08	478	485.86	296	330.14	<b>257</b>	<b>276.60</b>	472	491.84
Average	308.90	316.38	310.50	315.75	239.20	262.26	<b>207.30</b>	<b>224.50</b>	307.30	317.47
p-value	6.71e-14									

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

212 Table 2 reports the comparative results between the best MA with OX2  
 213 (called  $MA_{OX2}$ ) and  $TS_{CB}$ , which is the state-of-art algorithm for CBP pre-  
 214 sented in [23]. Table 2 shows the same information as in Table 1, except for  
 215 the column “CC” which represents the difference between the best values found  
 216 by  $TS_{CB}$  and  $MA_{OX2}$ . A negative “CC” indicates a worse result of  $MA_{OX2}$   
 217 compared to  $TS_{CB}$ . It is clear that for the 20 test graphs,  $MA_{OX2}$  does not  
 218 compete well with  $TS_{CB}$ . Indeed,  $TS_{CB}$  is a powerful iterated tabu search algo-  
 219 rithm which uses three dedicated neighborhoods to effectively explore the search  
 220



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

Graph	$MA_{OX2}$		$TS_{CB}$		CC
	Best	Avg	Best	Avg	
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	<b>73.20</b>	<b>83.23</b>	
p-value	1.31e-4				

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

## 227 5 Understanding the performance differences of the 228 compared crossovers

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

### 234 5.1 Distance and Population Entropy

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

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

236 where  $d_{ij}$  is the distance between two solutions  $\gamma_i$  and  $\gamma_j$  of  $P$ , which is defined  
 237 as the number of the adjacent connections that are contained in  $\gamma_i$  but not in  $\gamma_j$ .  
 238 For example, given two solutions  $\gamma_1=\{h a b d e f c g\}$  and  $\gamma_2=\{b a c h g d f e\}$ .  
 239 The set of adjacent connections is  $\{ha, ab, bd, de, ef, fc, cg, gh\}$  for  $\gamma_1$  and  $\{ba,$   
 240  $ac, ch, hg, gd, df, fe, eb\}$  for  $\gamma_2$ . The common adjacent connections are  $\{ab, ef,$   
 241  $gh\}$  ( $ba$  and  $ab$  are the same connections). The distance  $d_{12}$  equals thus  $8-3=5$ .  
 242 This distance is used in [8] to deal with TSP whose solutions have the symmetry  
 243 feature. As shown in Fig. 1, CBP solutions have the feature of symmetry, so the  
 244 use of this distance measure is very important for CBP.

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

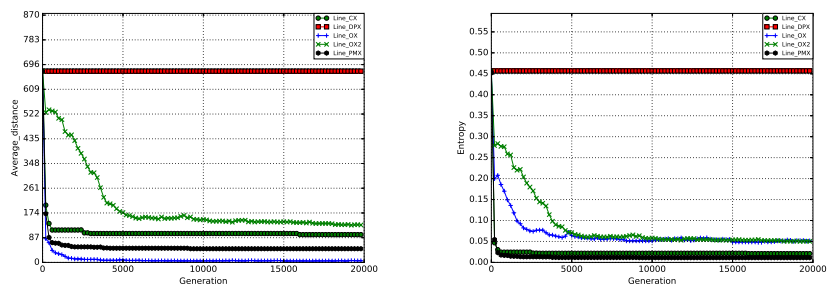
$$E_p(P) = \frac{-\sum_{i=1}^n \sum_{j=1}^n \left(\frac{n_{ij}}{|P|}\right) \log\left(\frac{n_{ij}}{|P|}\right)}{n \log n} \quad (6)$$

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

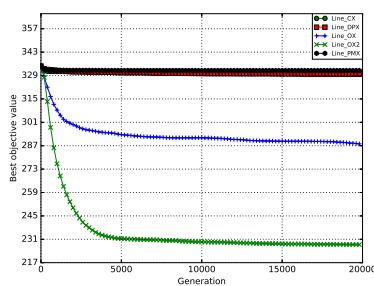
251 The instance ‘nos6’ is a representative large graph with 675 nodes from practical  
 252 application and rather difficult, so we use it as a working example. Figure  
 253 2 shows the average distance, average entropy and average best objective value  
 254 found in 50 independent executions over the graph ‘nos6’. Under 5000 gener-  
 255 ations, the population of the MA with OX2 has a high average distance and  
 256 entropy, leading to much better solutions. From generations 5000 to 20000, the  
 257 entropy is identical to that of OX, and the best average objective found stops  
 258 decreasing. These observations remain valid for all test graphs except the graph  
 259 ‘impcol.b’ (even if the MA with OX2 does not have a large population distance  
 260 and entropy, it gets good results comparing to others). Therefore, for the  
 261 operators CX, OX, OX2 and PMX, a higher entropy and average distance of  
 262 population leads to a good quality solution. However, what is surprising is that  
 263 the average distance and entropy with DPX always stay at a high level for all  
 264 test graphs, yet the quality of solutions found is not as good as that of the other  
 265 operators. To shed light on this behavior, we show a deeper analysis of the interaction  
 266 between the crossover mechanism and the characteristics of problem  
 267 in the next section.

## 268 5.2 Interaction between crossover and problem characteristics

269 In Section 5.1, we find that the recombination operator with a higher entropy  
 270 and average distance of the population generally helps to find solutions of good  
 271 quality. However, the DPX operator fails to reach good solutions even if the  
 272 entropy and average distance of population under the MA with DPX always stay  
 273 at a high level. From Figure 3, which presents the average objective value of the  
 274 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.

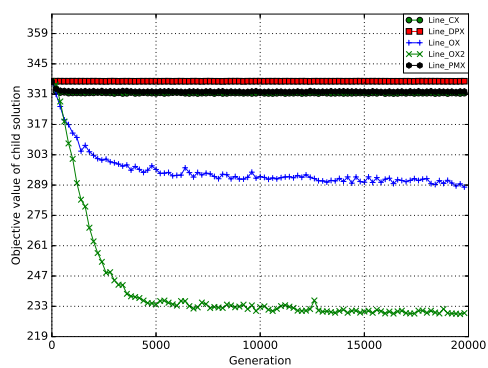


Fig. 3. Average objective value of the child solution over 50 independent executions.

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

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

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

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

## 298 6 Conclusion

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

318 **Acknowledgments**

319 We are grateful to the referees for their valuable suggestions and comments which  
 320 helped us to improve the paper. Support from the China Scholarship Council  
 321 (CSC, Grant 201608070103) for the first author and support from the Mexican  
 322 Secretariat of Public Education through SEP-CINVESTAV (2019–2020, Grant  
 323 00114) for the third author are also acknowledged.

324 **References**

- 325 1. Bansal, R., Srivastava, K.: A memetic algorithm for the cyclic antibandwidth max-  
 326 imization problem. *Soft Computing* **15**(2), 397–412 (2011)
- 327 2. Benlic, U., Hao, J.K.: Memetic search for the quadratic assignment problem. *Ex-  
 328 pert Systems with Applications* **42**(1), 584–595 (2015)
- 329 3. Bhatt, S.N., Leighton, F.T.: A framework for solving VLSI graph layout problems.  
 330 *Journal of Computer and System Sciences* **28**(2), 300–343 (1984)
- 331 4. Boese, K.D.: Cost versus distance in the traveling salesman problem. UCLA Com-  
 332 puter Science Department Los Angeles (1995)
- 333 5. Chen, Y., Hao, J.K.: Memetic search for the generalized quadratic multiple knap-  
 334 sack problem. *IEEE Transactions on Evolutionary Computation* **20**(6), 908–923  
 335 (2016)
- 336 6. Davis, L.: Applying adaptive algorithms to epistatic domains. In: *International  
 337 Joint Conference on Artificial Intelligence*. vol. 85, pp. 162–164 (1985)
- 338 7. Fleurent, C., A. Ferland, J.: Object-oriented implementation of heuristic search  
 339 methods for graph coloring. *Cliques, Coloring, and Satisfiability*. DIMACS Series  
 340 in Discrete Mathematics and Theoretical Computer Science **26**, 619–652 (1996)
- 341 8. Freisleben, B., Merz, P.: A genetic local search algorithm for solving symmetric and  
 342 asymmetric traveling salesman problems. In: *Proceedings of IEEE International  
 343 Conference on Evolutionary Computation*. pp. 616–621. IEEE (1996)
- 344 9. Goldberg, D.E., Lingle, R., et al.: Alleles, loci, and the traveling salesman prob-  
 345 lem. In: *Proceedings of International Conference on Genetic Algorithms and Their  
 346 Applications*. Lawrence Erlbaum, Hillsdale, NJ, vol. 154, pp. 154–159 (1985)
- 347 10. Hao, J.K.: Memetic Algorithms in Discrete Optimization. In F. Neri, C. Cotta,  
 348 P. Moscato (Eds.) *Handbook of Memetic Algorithms*. Studies in Computational  
 349 Intelligence 379, Chapter 6, pp. 73–94 (2012)
- 350 11. Jin, Y., Hao, J.K., Hamiez, J.P.: A memetic algorithm for the minimum sum  
 351 coloring problem. *Computers & Operations Research* **43**, 318–327 (2014)
- 352 12. Krasnogor, N., Smith, J.: A tutorial for competent memetic algorithms: model,  
 353 taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*  
 354 **9**(5), 474–488 (2005)
- 355 13. Lai, X., Hao, J.K.: A tabu search based memetic algorithm for the max-mean  
 356 dispersion problem. *Computers & Operations Research* **72**, 118–127 (2016)
- 357 14. Leung, J.Y., Vornberger, O., Witthoff, J.D.: On some variants of the bandwidth  
 358 minimization problem. *SIAM Journal on Computing* **13**(3), 650–667 (1984)
- 359 15. Lin, Y.: The cyclic bandwidth problem. *Chinese Science Abstracts Series A* **14**(2  
 360 Part A), 14 (1995)
- 361 16. Merz, P., Freisleben, B.: Memetic algorithms for the traveling salesman problem.  
 362 *Complex Systems* **13**, 297–345 (1997)

- 363 17. Moscato, P., Cotta, C.: A Gentle Introduction to Memetic Algorithms, pp. 105–  
364 144. Springer US, Boston, MA (2003)
- 365 18. Oliver, I., Smith, D., Holland, J.: A study of permutation crossover operators on  
366 the travelling salesman problem, genetic algorithms and their applications. In:  
367 Proceedings of the updating pop Second International Conference on Genetic Al-  
368 gorithms. pp. 224–230 (1987)
- 369 19. Ren, J, Hao, J.K., Rodriguez-Tello, E.: An iterated three-phase search approach  
370 for solving the Cyclic Bandwidth Problem. *IEEE Access* **7**, 98436–9845 (2019)
- 371 20. Rodriguez-Tello, E., Betancourt, L.C.: An improved memetic algorithm for the  
372 antibandwidth problem. In: International Conference on Artificial Evolution (Evo-  
373 lution Artificielle). pp. 121–132. Springer (2011)
- 374 21. Rodriguez-Tello, E., Hao, J.K., Torres-Jimenez, J.: An improved simulated an-  
375 nealing algorithm for bandwidth minimization. *European Journal of Operational*  
376 *Research* **185**(3), 1319–1335 (2008)
- 377 22. Rodriguez-Tello, E., Narvaez-Teran, V., Lardeux, F.: Comparative study of differ-  
378 ent memetic algorithm configurations for the cyclic bandwidth sum problem. *Paral-  
379 lell Problem Solving from Nature, PPSN XV*. pp. 82–94. Springer, Cham (2018)
- 380 23. Rodriguez-Tello, E., Romero-Monsivais, H., Ramirez-Torres, G., Lardeux, F.: Tabu  
381 search for the cyclic bandwidth problem. *Computers & Operations Research* **57**,  
382 17–32 (2015)
- 383 24. Romero-Monsivais, H., Rodriguez-Tello, E., Ramírez, G.: A new branch and bound  
384 algorithm for the cyclic bandwidth problem. *Advances in Computational Intelli-  
385 gence. MICAI 2012. Lecture Notes in Computer Science*, vol 7630. Springer, Berlin,  
386 Heidelberg. pp. 139–150 (2012)
- 387 25. Rosenberg, A.L., Snyder, L.: Bounds on the costs of data encodings. *Mathematical*  
388 *Systems Theory* **12**(1), 9–39 (1978)
- 389 26. Syswerda, G.: Scheduling optimization using genetic algorithms. *Handbook of Ge-  
390 netic Algorithms*. pp. 322–349 (1991)
- 391 27. Wang, Y., Lü, Z., Hao, J.K.: A study of multi-parent crossover operators in a  
392 memetic algorithm. *Parallel Problem Solving from Nature, PPSN XI. LNCS*, vol  
393 6238. Springer, Berlin, Heidelberg, pp. 556–565 Springer (2010)
- 394 28. Wu, Q., Hao, J.K.: Memetic search for the max-bisection problem. *Computers &*  
395 *Operations Research* **40**(1), 166–179 (2013)
- 396 29. Zhou, Y., Hao, J., Glover, F.: Memetic search for identifying critical nodes in sparse  
397 graphs. *IEEE Transactions on Cybernetics* **49**(10): 3699–3712 (2019)