Memetic Algorithm with Hungarian Matching Based Crossover and Diversity Preservation

Emmanuel Romero Ruiz\textsuperscript{1}, Carlos Segura\textsuperscript{2}

\textsuperscript{1} Universidad de Guanajuato, Guanajuato, Mexico
\textsuperscript{2} Centro de Investigación en Matemáticas (CIMAT), Área de Computación, Guanajuato, Mexico
\{emmanuel.romero, carlos.segura\}@cimat.mx

Abstract. The Graph Partitioning Problem (GPP) is a well-known NP-hard combinatorial problem that involves the finding of a partition of vertexes that minimizes the number of cut edges while fulfilling a set of constraints. This paper presents a newly designed optimizer for the GPP: the Memetic Algorithm with Hungarian Matching Based Crossover and Diversity Preservation (MAHMBCDP). MAHMBCDP is a population-based scheme that incorporates an explicit mechanism to control the diversity with the aim of making a proper use of resources when dealing with long-term executions. Among the novelties of our proposal, the design of a crossover operator that is based on the Hungarian Algorithm to calculate a maximum matching is particularly important. Experimental validation with a set of well-known instances of the graph partitioning archive shows the proper performance of our proposal. In fact, new best-known solutions could be attained in ten test cases.

Keywords. Graph partitioning problem, memetic algorithm, diversity preservation, maximum matching.

1 Introduction

The Graph Partitioning Problem (GPP), is a well-known NP-hard [10], combinatorial optimization problem that has been addressed with a large number of different techniques [4]. One of the reasons why the GPP is probably so popular is because it arises in several practical cases [6]. For instance, it has been used for the mesh partitioning in simulators that apply the finite element and/or the finite volume paradigms, in the implementation of parallel software, and in the design of circuit layouts, among others.

Solving the GPP means finding a partition of the vertexes of a given graph that fulfill a balance constraint among the sizes of the different subsets and at the same time minimizes the number of cut edges, i.e. the number of edges that run between vertexes of different subsets. Note that, in the related literature, the terms subset, parts and classes are used to refer to each one of the groups of vertexes established by the partition. Thus, any of these concepts are used indifferently in this paper.

In many of the applications of the GPP quite large graphs arise [14]. Given the complexity class of the GPP, exact solvers, such as some based on quadratic programming formulations [11], are nowadays not applicable to practical graphs. As a result, several heuristics and metaheuristics have been proposed with the aim of obtaining high-quality approximate solutions for such large cases. Initially, several constructive and improvement heuristics were devised [15]. While they are fast, the quality of the obtained partitions is not so large. Subsequently, trajectory-based metaheuristics were adapted to deal with the GPP [3], reaching much higher quality solutions at the cost of an increase in the
required computational budget. Finally, nowadays hybrid population-based metaheuristics lead the attainment of high-quality solutions. In fact, most of the best-known solutions that have been attained for the instances that are maintained in the graph partitioning archive [13], have been obtained with population-based approaches. Particularly, memetic algorithms that combine an Evolutionary Algorithm (EA) with local search or with trajectory-based metaheuristics seem to be the most promising approaches [8].

There are several problems that are closely related to the GPP. One of them is the Frequency Assignment Problem (FAP). In the FAP, a set of partitions must also be identified. However, the objective is related to minimizing the number of edges that appear between vertexes that belong to the same class, instead of vertexes that belong to different classes [1]. In any case, some of the ideas that are used in the optimization of the FAP can be applied to the GPP. In a recent research [25] a novel memetic algorithm could provide an important advance in the solving of the FAP. In fact, several new best-known solutions could be attained for some well-known benchmarks. The most important novelty of the solver was the application of a mechanism to explicitly control the diversity of the population. Particularly, the main aim is to lose the diversity in a controlled way that depends on the stopping criterion set by the user. In this way, shorter stopping criteria induce a faster convergence.

The hypothesis behind the development of this research is that the kinds of mechanisms that provide benefits in the FAP might also help in the development of optimizers for the GPP. We based our hypothesis on two aspects. First, on the similarities that exist between the current optimizers for the FAP and GPP. Second, in the fact that state-of-the-art population-based approaches reach convergence of the population in relatively few generations. For instance, in [8], only 200 generations are evolved because after such number of generations all the individuals of the population are located in a similar region and no further improvements are obtained with larger executions. Since the appearance of premature convergence in typical EAs seems to be a key for the success of the schemes that control the diversity in an explicit way, applying variants of this proposal seems truly promising. Note that another particular feature of the method developed in [25], is that it applies a computationally expensive local search with a large neighborhood and ad-hoc crossover operators. As a result, the memetic algorithm developed in this paper also considers a hill climber with a large neighborhood [22] and a newly designed crossover operator adapted to the GPP.

Experimental validation has been performed with a set of test cases of the well-known graph partitioning archive. Particularly, 15 graphs with up to 16,840 nodes have been taken into account. The partitions obtained with these graphs for different number of subsets show the proper performance of our proposal. In fact, new best-known solutions could be obtained in ten test cases. The most important advances appear in not too large graphs and with few subsets. The increase in the number of edges, vertexes and/or subsets implies a drastic decrease in the number of generations that can be evolved, meaning that the advantages of controlling the diversity in an explicit way diminish.

The rest of the paper is organized as follows. The mathematical formulation of the GPP is presented in Section 2. Related work is discussed in Section 3. Our novel proposals are detailed in Section 4. Section 5, is devoted to our experimental validation. Finally, our conclusions and future work are presented in Section 6.

2 Graph Partitioning Problem: Mathematical Definition

This section provides a formal definition of the GPP [2]:

— Let \( G = (V, E, w) \), be an undirected graph where \( V \) is the set of nodes, \( E \) the set of edges and \( w : E \rightarrow \mathbb{R}_{>0} \), a weight function.

— In the following \( n = ||V|| \) and \( m = ||E|| \).

— Let \( k \) be a fixed positive integer that represents the number of classes in the desired partition and \( \epsilon > 0 \) a real number that is related to the balance constraint.
— The feasible search space for the problem is the set of partitions \( \Delta = \{ S = \{ S_1, S_2, ..., S_k \} \) such that \( S \) is a partition into \( k \) sets of \( V \) and \( \| S_i \| \leq L_{max} := (1 + \epsilon) \lceil n/k \rceil \) for all \( i \in \{1, 2, ..., k \} \).

— Finally the function to minimize in the GPP is:
\[
f(S) = \sum_{i<j} w(E_{ij}) \text{ where } E_{ij} = \{(u, v) \in E : u \in S_i, v \in S_j \} \text{ and } w(C) = \sum_{e \in C} w(e).
\]

The experimental validation of this paper takes into account the instances present in the Chris Walshaw’s Graph Partitioning Archive [13]. In these instances, all the weights are equal to 1, so the objective is to minimize the number of conflicting edges. Moreover, the archive reports results for distinct values of \( \epsilon \). However, this paper focuses in the case where \( \epsilon = 0 \). Some minor modifications are required to deal with different \( \epsilon \) values. However, performing these adaptations is out of the scope of this work.

3 Related Work

This section is devoted to reviewing some of the works that are closely related to our approaches. First, among the large set of EAs that have been used to tackle the GPP, some of the ones that have obtained the most promising results, as well as the ones that share several features with our proposal are discussed. In addition, since our proposal is based on controlling the diversity explicitly, the most popular techniques that have been devised with the aim of alleviating the effects of premature convergence are summarized. Some of these techniques are used to validate our proposal.

3.1 EAs for the GPP

A large amount of population-based metaheuristics has been designed to deal with the GPP [4]. Among them, EAs are probably the most popular approaches. Since the initial proposals [5], it was clear that incorporating a procedure to intensify in the regions located by the EA was quite important. In most of the initial algorithms, the heuristic proposed by Kernighan and Li was taken into account [15].

This heuristic is an inexpensive process that attains competitive partitions. More complex ways of intensifying have been defined with the increase of the computational power. In some cases, EAs are integrated with trajectory-based strategies [8], whereas in other ones, hill-climbers with larger neighborhoods are applied [22]. These kinds of approaches imply the use of additional computational resources, but they have allowed improving further the best-known solutions in several large graphs. Thus, in our proposal, the relatively complex neighborhood defined in [22], is taken into account.

The design of proper crossover operators has also implied large research efforts. In the initial approaches, general operators such as the uniform and/or the n-point crossover were applied [17]. However, it was soon clear that these operators were quite disruptive. Particularly, they had a tendency to create quite unbalanced partitions, meaning that after the balancing mechanisms, the offspring might share just a few features with their parents [4]. An attempt to avoid this issue was devised in [6]. In this case, the vertexes that share the same class in both parents maintain their class in the offspring.

Then, the remaining vertexes are set by using a greedy constructive heuristic that ensures a proper balance. One fact that is not taken into account in the previous operator is that the specific class identifier of a vertex is not really important. The important features are related to the set of vertexes that share the same class. Taking this into account, more complex crossover operators that are based on inheriting these kinds of features have been devised [9, 8]. The principle behind these schemes is to transmit large sets of vertexes that share the same class between parents and offspring. In [8], a greedy approach based on maximizing intersections is used. More complex non-greedy proposals were previously suggested in [9]. However, in this last case, it was applied to the graph coloring problem and not to the GPP. Since these kinds of operators have reported really promising results, our proposal also takes this principle into account. Particularly, some of the ideas suggested in [9], inspired the development of our novel crossover operator.
However, our operator also induces the creation of relatively balanced partitions in the offspring, which is an important feature in the GPP but not in the graph coloring problem.

Finally, an important issue that has not been studied in depth is the management of diversity. However, some of the analyses show that the proper control of diversity might be an issue for the proper performance. In [8], the number of generations was limited to 200. Authors argue that after this number of generations, all the individuals are located in the same region and that further improvements are not expected with larger executions. Additionally, in many cases, some actions to delay the convergence are included. For instance, in [5] offspring substitute the most similar parent, whereas resorting to restarting mechanisms when convergence is detected is another typical approach [19].

### 3.2 Control of Diversity in EAs

Since our approach incorporates a novel way of controlling the diversity, this subsection discusses some of the most popular strategies that have been designed with the aim of avoiding premature convergence. Particularly, the techniques that have been used in the experimental validation of our proposal are described. Readers are referred to [28] for additional techniques and for more complete descriptions. Note that, a large number of techniques to deal with premature convergence have been devised [20].

The methods are usually classified in base of the component of the EA that they modify in the following groups [28]: selection-based, population-based, crossover/mutation-based, fitness-based, and replacement-based. Additionally, depending on the number of altered components, they are referred to as uniprocess-driven or multiprocess-driven. Although the development of multiprocess-driven schemes might result in higher-quality solutions, currently most of the efforts have been placed on the design of uniprocess-driven strategies. Controlling several related stochastic components simultaneously is much more complex, so probably for this reason multiprocess-driven schemes are not so popular. As a result, our proposal is a uniprocess-driven strategy. Among the different strategies, the replacement-based methods have reported quite promising results with several optimization problems [26, 23]. Since our novel proposal belongs to this category, three additional replacement-based schemes are taken into account in this paper:

- The Restricted Tournament Selection [12], (RTS) is a popular steady-state scheme in which after each new individual \(O\) is created, \(CF\) individuals from the current population are selected at random. Then, the best individual between \(O\) and its most similar individual in the selected set survives. Note that this scheme belongs to the group of crowding strategies [18].

- In the Hybrid Genetic Search with Adaptive Diversity Control [29] (HGSADC), individuals are sorted by their contribution to diversity and by their original cost. Then, the rankings of the individuals are used to calculate a score using two parameters \(N_{Close}\) and \(N_{Elite}\) which are used to determine the survivors.

- The clearing strategy (CLR) [21] is an extension of fitness sharing and it alters both the fitness assignment procedure and the replacement selection phase. In the clearing procedure, individuals are grouped into niches — defined via the parameter \(\sigma\) — and the resources of a niche are attributed to the best \(W\) elements in each niche. Moreover, the winners of each niche are copied to the next population. In order to avoid a large immobilization of the population, the winners with a fitness lower than the mean are discarded.

Note that, in addition to these techniques, a generational scheme with elitism [7], \(\text{GEN}_\text{ELIT}\) is also incorporated in our experimental validation. This replacement phase does not incorporate any special mechanism to delay convergence. However, since it is one of the mostly applied strategies, it was taken into account. In \(\text{GEN}_\text{ELIT}\) the new population contains the offspring and the best solution of the previous generation.
4 Proposal

Our proposal (MAHMBCDP) is an EA that incorporates a local search strategy. The general algorithm is a quite standard Memetic Algorithm (see Algorithm 1). The only modification to the general approach is the incorporation of a balancing strategy to fulfill the GPP constraints. In order to fully define our proposal, the genetic operators, balancing operator, local search and replacement strategy must be described. The next subsections details each of these components.

Algorithm 1 Diversity-based Lamarckian Memetic Algorithm
1: **Initialization:** Generate an initial population $P_b$ with $N$ individuals. Assign $t = 0$.
2: **Local Search:** Perform a local search for every individual in the population.
3: while (not stopping criterion) do
4: **Mating selection:** Perform binary tournament selection on $P_t$ in order to fill the mating pool.
5: **Variation:** Apply genetic operators to the mating pool to create a child population $C_P$.
6: **Balancing** Apply a balancing phase because of the problem's balance restrictions.
7: **Local Search:** Perform a local search for every individual in the offspring.
8: **Survivor selection:** Combine $P_t$ and $C_P$, and apply the BNP survivor selection strategy.
9: $t = t + 1$
10: end while

4.1 Crossover Operator

As usual, the crossover operator takes two individuals and generates two new ones with some similar features to the parents. Many of the crossover operators that are used for the GPP are inspired in those developed for the graph coloring problem. Our proposal considers a newly designed crossover operator, the Hungarian Based Crossover (HBX) — see Algorithm 2 — that is inspired by the one devised in [9]. Particularly, it is based on the principle of maximizing the number of nodes that share the same class both in the parents and offspring.

The HBX operator works as follows. Let $I_1 = \{S_{11}, S_{12}, ..., S_{1k}\}$ and $I_2 = \{S_{21}, S_{22}, ..., S_{2k}\}$ be the parents selected for the crossover operator. The Hungarian method is a well-known combinatorial optimization algorithm that solves, in polynomial time, the assignment problem in weighted bipartite graphs, i.e. it calculates the maximum or minimum weight matching. In this case, a complete bipartite graph $(G)$ with $2 \times k$ vertexes is built. The first $k$ vertexes are associated to the subsets of $I_1$, whereas the last ones correspond the subsets of $I_2$. The edges connect any of the first $k$ vertexes with any of the last $k$ vertexes. The weight of each edge is established as the size of the intersection of the sets associated with each vertex.

Algorithm 2 Hungarian Based Crossover
1: Let $elg, r$ a boolean vector for the "eligible" rows.
2: Let $elg, c$ a boolean vector for the "eligible" columns.
3: Let $blc, r$ a boolean vector for the "blocked" rows.
4: Let $blc, c$ a boolean vector for the "blocked" columns.
5: The elg vectors are set to true.
6: The blc vectors are set to false.
7: for $i$ in $\{1, 2, ..., n\}$ do
8: if $i \equiv 0$, mod(2) then
9: $L = NULL$
10: for $i$ such that $elg, c[i]$ is true (in random order) do
11: $B_i = \bigcup_{k=1}^{m} \{ P_{k,l} \text{ not erased} \} \cup \{ P_{k,l} \text{ such that } blc, c[k] \text{ is true } \}$
12: if $L == NULL$ or $|B_i| > |B_L|$ set $L = i$
13: end for
14: $T_i = B_L$
15: Erase from $P$ all the elements of $T_i$
16: $elg, c[L] = false$
17: $elg, r[\sigma^{-1}(L)] = false$
18: $blc, c[\sigma^{-1}(L)] = true$
19: else
20: $L = NULL$
21: for $i$ such that $elg, r[i]$ is true (in random order) do
22: $B_i = \bigcup_{k=1}^{m} \{ P_{l,k} \text{ not erased} \} \cup \{ m, P_{o(l)} \text{ such that } blc, r[m] \text{ is true } \}$
23: if $L == NULL$ or $|B_i| > |B_L|$ set $L = i$
24: end for
25: $T_i = B_L$
26: Erase from $P$ all the elements of $T_i$
27: $elg, r[L] = false$
28: $elg, c[\sigma(L)] = false$
29: $blc, c[\sigma(L)] = true$
30: end if
31: end for
The permutation $\sigma$ obtained with the application of the Hungarian Algorithm to $G$ maximizes:

$$\sum_{i=1}^{k} \| S^1_i \cap S^2_{\sigma(i)} \|.$$  

**4.1.1 Main Properties of HBX**

Previously to giving the specific crossover procedure, it is important to discuss some of the properties that share the offspring. Let $P$ be an intersection matrix defined in a way that $P_{i,j} = S^1_i \cap S^2_j$, and let $T = \{ T_1, T_2, ..., T_k \}$ be the offspring. Then, $T$ has the following properties:

1. For each $j$, there is a unique $i_j$ such that $S^1_{i_j} \cap S^2_{\sigma(i_j)} \subset T_j \subset S^1_{i_j} \cup S^2_{\sigma(i_j)}$.
2. $P_{i,j} \subset T_l$ for all the pairs $(i,j)$ and some $l$.
3. $T_l$ is the union of exactly $k$ distinct $P_{i,j}$.

The reason behind the use of $\sigma$ is that the Hungarian Algorithm gives the maximum matching between the sets in both partitions. This means that $S^1_i$ is somewhat similar to $S^2_{\sigma(i)}$ and their intersection is susceptible to have a lot of elements. The property 1) establishes that these intersections are used as the core of the sets in the new partition. It also establishes that the elements are selected exclusively from the unions of the considered subsets.

Taking these properties into account, it can be stated that operators that share these properties are not very destructive operators. This because property 2) focuses on inheriting together large subsets of vertexes that were assigned to the same subset in both parents. Finally, the principle behind the third property, is to avoid an excessive imbalance in the sizes of the newly generated subsets. However, note that some imbalance usually appears, so a method to properly balance the subsets is required.
Algorithm 3 Mutation based on Random Connected Component

1: Let \( w \) be a random node in the graph.
2: Let \( S \) be a set that contains \( w \).
3: \( k = n_{iter} \)
4: while \( k > 0 \) do
5: for \( u \) in \( S \) do
6: for \( v \) neighbor of \( u \) do
7: Take \( p \) random in \([0, 1]\).
8: if \( p < p_m \) then
9: Add \( v \) to \( S \)
10: end if
11: end for
12: end for
13: \( k = - \)
14: end while
15: Move all the vertexes in \( S \) to a randomly selected subset

4.1.2 Construction

This section is devoted to present the specific algorithm used to build two offspring that fulfill the aforementioned properties. Let \( P \) be the intersection matrix previously defined. Then Algorithm 2 is the process required to create the first new individual. This process is illustrated in Fig. 1. The main principle is to select in each step the position \((i, \sigma(i))\) of the matrix, where the union of some elements of the corresponding row and column (see lines 11 and 22 for details) is maximized. Note that it can be proved that in line 11 there are exactly \( k - j/2 \) distinct \( P_{i,j} \) not erased from \( P \) and \( j/2 P_{\sigma^{-1}(i),m} \) such that the column \( m \) is “blocked”. Something similar is true in line 22 and this proves the property 3 of HBX.

In order to generate the second offspring, the order in which rows and columns are considered is exchanged, i.e. the even iterations take into account the columns, whereas the odd iterations take into account the rows. Note that regardless of the order considered, both individuals share the same properties described above.

In Fig. 1 the evolution of HBX for a specific individual is shown. In this case, \( k = 4, \sigma = (2, 1, 3, 4) \). Each of the four images in Fig. 1 can be considered as a matrix, \( P \). Each entry, \( P_{i,j} \), of these matrices represent the intersection of the corresponding sets in the parents \( S^1, S^2 \).

In the last image, the union of the sets with the same color represents an element in the new partition, \( T \). The first image shows how \( T_1 \) is chosen as a row from \( P \) and how column 3 is blocked. Note that the row was chosen randomly and the column considered the Hungarian permutation. In the second image, part of \( T_2 \) is taken from a column of \( P \) and the remaining elements are taken from sets in row 2 that are in a blocked column \((P_{2,3})\). Similarly in the image 3 and 4, \( T_3 \) and \( T_4 \) are created. Everything follows the procedure described in Algorithm 2.

Another important fact of the novel operator is that no information belonging to the GPP definition was used in the design. Thus, this crossover operator might be used in any optimization problem in which sharing the same value in different variables is an important feature.

4.2 Mutation Operator

In the GPP it is desirable to keep connected components together in one subset of the partition. The reason is that connected components encapsulates common edges, meaning that lower cut edges might be induced. Taking this property into account, the principle of the mutation operator is to move parts of connected components from one subset to another one. In order to select this subcomponent, the same approach than the one presented in [25] is applied. Then, the selected vertexes are moved to a set \( S_i \), where \( i \) is selected in a random way. Algorithm 3 describes the process. In this work we use \( p_m = 0.1 \) and \( n_{iter} = 5 \).

4.3 Balancing Operator

The balancing routine consists of two phases and it is very important because of the problem’s balancing restriction. This restriction might not be valid after the application of the crossover and/or mutation operator so it has to be restored. The procedure that is in charge of this process is given in Algorithm 4.

In the first phase, nodes are moved from larger to smaller sets by taking into account the implications of the move on the edge cut. This process
Algorithm 4 Balancing Operator

1: Let ind the input individual to balance.
2: \( k = n_{iterBal} \)
3: while \( k > 0 \) and ind is not balanced do
4:   Select \( i \) randomly in \( \{i \text{ such that } \|S_i\| \text{ is not biggest set of ind}\} \).
5:   Find \( u \) the node of maximum gain among \( \cup S_j \) such that \( \|S_j\| > \|S_i\| \).
6:   Move \( u \) to \( S_i \).
7:   \( k = k - 1 \)
8: end while
9: while ind is not balanced do
10:   Choose \( u \in S_j \) such that \( \|S_j\| > S_{\text{max}} \).
11:   Move \( u \) to a random set \( S_k \) with \( \|S_k\| < S_{\text{max}} \).
12: end while

is repeated \( n_{iterBal} \), which is set to \( |V| \) in our experimental validation.

The second phase, which ensures the attainment of a properly balanced solution, moves randomly selected nodes from sets that do not fulfill the given restriction to subsets where there is space available.

4.4 Local Search

In order to create individuals with better fitness, a local search algorithm was added. This routine is performed after the balancing procedure and it consists of two phases.

In the first phase, the strategy that is called the “perfectly balanced local search by negative cycle detection” in [22] is carried out. The principle behind this algorithm is to find a cycle of negative gain and move the nodes around it. In this way the balance is kept and the fitness is diminished.

The second phase is a stochastic hill climbing. For this routine two individuals are neighbor if their only difference is a swap in an edge. It means, if \((u, v)\) is an edge and individual \(I_1\) fulfill that \(u \in S_{c_u} \) and \(v \in S_{c_v}\), with \(c_u \neq c_v\), then individual \(I_2\) such that \(u \in S_{c_v}\) and \(v \in S_{c_u}\) is neighbor of \(I_1\) if and only if the previous one is their only difference. Once the notion of neighborhood is defined, a usual stochastic hill climbing is applied [27].

Algorithm 5 BNP survivor selection technique

Require: Population, Offspring

1: for all \( I \in \text{Offspring} \) do
2:   I.cost = edge cut associated with individual \( I \)
3: end for
4: Penalized = \( \emptyset \)
5: CurrentIndividuals = Population \( \cup \) Offspring
6: Best = Individual with lowest edge cut in CurrentIndividuals
7: NewPop = \{ Best \}
8: CurrentIndividuals = CurrentIndividuals \( \{ \text{Best} \} \)
9: Update \( D \) taking into account the elapsed time \((T_e)\), stopping criterion \((T_s)\) and initial value of \( D \) \((D_1)\): 
10: \( D = D_1 - D_1 \ast \frac{t}{T_e} \). We took \( D_1 = \text{distInitFactor} \ast M \), where \( M \) is the mean distance in the first population and \(\text{distInitFactor} \in [0, 1] \).
11: while \((|\text{NewPop}| < N)\) do
12:   for all \( I \in \text{CurrentIndividuals} \) do
13:     I.DCN = distance to the closest individual of \( I \) in \( \text{NewPop} \)
14:     if \( I.DCN < D \) then
15:       I.cost = Infinity
16:       Penalized.insert(I)
17:       CurrentIndividuals.erase(I)
18:       end if
19:     end for
20:     for all \( I \in \text{Penalized} \) do
21:       I.DCN = distance to the closest individual of \( I \) in \( \text{NewPop} \)
22:     for all \( I \in \text{CurrentIndividuals} \) do
23:       Selected = I with largest I.DCN in \( \text{Penalized} \)
24:       Penalized = Penalized \( \{ \text{Selected} \} \)
25:     else
26:       Selected = I with lowest I.cost in CurrentIndividuals
27:       CurrentIndividuals = CurrentIndividuals \( \{ \text{Selected} \} \)
28:     end if
29:     NewPop = NewPop \( \cup \) Selected
30: end while
31: Population = NewPop

4.5 Survivor Selection

Our memetic algorithm applies a novel replacement phase which is called the Best-Non-Penalized (BNP) survivor selection strategy (see Algorithm 5). One of the principles of the BNP strategy is to avoid the selection of too close individuals. Specifically, the approach tries to avoid
the selection of pairs of individuals that are closer than a $D$ value. Since in the initial phases it is important to explore, whereas in the last phases the process should intensify, the $D$ value varies during the execution. Particularly, the variable $D$ is initialized in base of the content of the first population (see line 9) and it is decreased in a linear way.

In each execution of the survivor operator $N$ individuals from the previous population and offspring are selected to survive. BNP iteratively selects the best element that is not penalized. Then, all the remaining individuals with distance at most $D$ to the previously selected ones are penalized. If it is not possible to find a non-penalized individual, the individual with larger distance to the currently selected individuals is taken.

Note that the principles of BNP are similar to the ones that guided the design of the Replacement with Multi-objective based Dynamic Diversity Control strategy (RMDDC) [25]. The main difference is that in RMDDC the same importance is given to quality and diversity, so a multi-objective selection is used. In the case of GPP, due to the high computational cost associated with some components, not too many generations can be evolved. Thus, the incorporation of an additional bias towards high-quality solutions that is performed in the BNP strategy is mandatory to obtain high-quality solutions with the stopping criterion set in our analyses.

An important fact that should be noticed is that the survivor operator requires a distance-like function between individuals. In this case, a function based on the Hungarian Algorithm is used. Let $\tau$ be a permutation of $k$ elements, then

$$d(S^1, S^2, \tau) = |V| - \sum_{i=1}^{k} \|S^1_i \cap S^2_{\tau(i)}\|.$$ 

Our distance is defined as the minimum $d(S^1, S^2, \tau)$ over all the $\tau$'s. This value can be calculated efficiently by using the Hungarian Best Matching Algorithm.

### 5 Experimental Validation

In this section, the experiments conducted with our proposal (MAHMBCDP) are described. Since our proposal is a stochastic algorithm, each execution was repeated 30 times and comparisons were carried out by applying a set of statistical tests. A similar guideline as the one applied in [24] was used. Specifically, the following tests were applied, assuming a significance level of 5%. First, a Shapiro-Wilk test was performed to check whether or not the values of the results followed a Gaussian distribution.

If so, the Levene test was used to check for the homogeneity of the variances. If samples had equal variance, an ANOVA test was done; if not, a Welch test was performed. For non-Gaussian distributions, the non-parametric Kruskal-Wallis test was used to test whether samples are drawn from the same distribution. The following section describes the set of benchmarks and experiments that have been taken into account to validate our proposal.

Our approach involves the setting of some parameters. The population size was set to 50, the crossover probability ($p_c$) was set to 0.85 and the mutation probability ($p_m$) was set to 0.1. These values are quite standard values and we are aware that some improvements might be obtained by applying parameter setting schemes [16]. However, since very large executions were performed (the stopping criterion was set to 48 hours) and they provided promising results no further experimentation was carried out.

Our validation involves the application of several replacement strategies in addition to the BNP strategy used in MAHMBCDP. Depending on the replacement phase, additional parameters might be required. In order to set them, some initial experiments were developed by considering some values that have reported promising results [23] in other problems, as well as some modifications of these values. The ones that obtained the best performance are the following. In the RTS, $CF$ was set to 25. In HGSADC, $N_{Close}$ and $N_{Elit}$ were set to 3 and 8, respectively. In CLR, $\sigma$ was set to $0.20 \times M$, where $M$ is the mean distance among individuals of the first population.

Finally, in the BNP strategy, $distInitFactor$ was set to 0.6. More information on the sensitivity of this parameter is given in the following subsections.
Table 1. Features of the graphs selected for the experimental validation

| Graph    | $|V|$ | $|E|$ | Graph    | $|V|$ | $|E|$ |
|----------|------|------|----------|------|------|
| add20    | 2395 | 7462 | data     | 2861 | 15093 |
| 3elt     | 4720 | 13722| uk       | 4824 | 6837 |
| add32    | 4860 | 1402 | bcsstk33 | 8736 | 291583|
| whitakerd| 9800 | 28989| crack    | 10240| 30380 |
| wing,nodal| 10937| 75488| te,4elt2 | 1143 | 32818 |
| vibobox  | 12328| 165250| bcsstk29 | 13992| 302748|
| 4elt     | 15690| 49973| te,sphere| 16386| 49152 |

5.1 Benchmark

In order to evaluate the performance of our proposal, test cases from the graph partitioning archive (TGPA) of Chris Walshaw [13], were used. This is a site that has been maintained since 2000 and includes results from most of the important partitioning software packages. The whole set is composed of 34 graphs.

The smallest graph has 2,395 nodes and 7,462 edges and the biggest has 448,695 nodes and 3,314,611 edges. The number of edges in these graph are usually between two and twenty times the number of nodes. For each of these graphs, the best known partitions for $k \in \{2, 4, 8, 16, 32, 64\}$ and $\epsilon \in \{0.0, 0.01, 0.03, 0.05\}$ are available. The values $k \in \{4, 8, 16, 32, 64\}$ and $\epsilon = 0.0$ were taken into account for testing our algorithm. We chose $k = 2^n$ in order to compare our results with the ones in TGPA, but the scheme can be applied for any arbitrary $k$.

In this work, the first 15 graphs in TGPA were selected. The reason is that we identified that our algorithm was not able to evolve a reasonable number of generations (more than 1,000), in graphs with more than 17,000 edges, which is quite important for the proper performance of diversity-based memetic algorithms. Table 1 details the number of nodes and edges of the selected graphs.

5.2 Analysis of Diversity

One of the most important features of our approach is that it considers the diversity explicitly. Our proposal depends on a parameter, $distInitFactor \in [0, 1]$. In order to use the same parameter value in every case, a preliminary experiment was performed.

Particularly, five different equidistributed values were used with two instances and two different values of $k$. The mean and best results obtained for 30 executions are shown in Table 4 for each of the tested values. The values 0.6 and 0.8 reported quite competitive results. Since the value 0.6 reported a lower mean in more cases, this value is used in all the remaining experiments.

Table 2. Comparison among different schemes for the 3elt graph

<table>
<thead>
<tr>
<th>Scheme</th>
<th>$k = 4$</th>
<th>$k = 64$</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAHMBCDP</td>
<td>1533</td>
<td>1537.1</td>
</tr>
<tr>
<td>RTS</td>
<td>1536</td>
<td>1531.4</td>
</tr>
<tr>
<td>HGSADC</td>
<td>1536</td>
<td>1546.8</td>
</tr>
<tr>
<td>CLR</td>
<td>1545</td>
<td>1555.6</td>
</tr>
<tr>
<td>GEN_ELIT</td>
<td>1546</td>
<td>1555.6</td>
</tr>
</tbody>
</table>

For validating the BNP survivor selection integrated in MAHMBCDP, some additional popular schemes were also tested. Specifically, results were obtained with the following methods: RTS, HGSADC, CLR and GEN_ELIT. Table 2 summarizes the results obtained with the 3elt graph. Specifically, the best, mean and worst values are reported for $k = 4$ and $k = 64$. The same information is shown in Table 3 for the uk instance. It is clear that MAHMBCDP obtained the most promising results. In fact, the statistical tests described before were applied and showed the superiority of MAHMBCDP in every case. Particularly, the maximum p value was equal to 0.00013 in the comparisons.

In order to properly understand the reasons behind the superiority of the BNP technique it is important to analyze the evolution of the diversity and fitness (cut edge). Fig. 2 shows the evolution of the diversity whereas Fig. 3 shows the evolution of the fitness for the aforementioned cases. The diversity was measured as the mean distance among all the individuals in the population.

It is noticeable that the only scheme that induces a gradual decrease in diversity is MAHMBCDP, i.e. the scheme that applies the BNP strategy.
Fig. 2. Evolution of diversity in four different test cases

Fig. 3. Evolution of fitness in four different test cases
Table 3. Comparison among different schemes for the uk graph

<table>
<thead>
<tr>
<th>Scheme</th>
<th>k = 4</th>
<th>k = 64</th>
<th>k = 4</th>
<th>k = 64</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best</td>
<td>Mean</td>
<td>Worst</td>
<td>Best</td>
</tr>
<tr>
<td>MAHMBCDP</td>
<td>41</td>
<td>45.8</td>
<td>73</td>
<td>420</td>
</tr>
<tr>
<td>RTS</td>
<td>153</td>
<td>198.73</td>
<td>249</td>
<td>444</td>
</tr>
<tr>
<td>HGSADC</td>
<td>119</td>
<td>187.13</td>
<td>249</td>
<td>450</td>
</tr>
<tr>
<td>CLR</td>
<td>195</td>
<td>253.2</td>
<td>329</td>
<td>585</td>
</tr>
<tr>
<td>GEN_ELIT</td>
<td>194</td>
<td>247.5</td>
<td>322</td>
<td>466</td>
</tr>
</tbody>
</table>

The large diversity in the initial phases explains why MAHMBCDP does not attain very promising partitions initially. In this phase, it is looking for promising regions in the variable space but intensification is not promoted. At the end of the execution, the diversity is low, meaning that intensification is promoted in the most promising regions identified. This results in solutions of higher quality in the long term.

An interesting thing is that the decrease of diversity was not linear as we expected by decreasing $D$ in Algorithm 5 in that way.

This is specially clear in the cases with $k = 4$. Thus, creating an adaptive algorithm to manage the decrease of diversity seems promising, but that work is beyond the reach of this paper.

5.3 Validation with the Graph Partitioning Archive

In order to illustrate the effectiveness of the proposed memetic algorithm, test with the 15 first graphs of the Graph Partitioning Archive were performed. MAHMBCDP was tested with the values $k = \{4, 8, 16, 32, 64\}$. Tables 5, 7 and 6 summarize the results obtained with MAHMBCDP. In addition to the best, mean and worst results obtained, the best-known solution found by any solver previous to this paper is shown in the column TGPA. MAHMBCDP reaches the best known solution for almost every instance for $k = 4$ and $k = 8$. Moreover, in five test cases it could generate a new best-known solution. The only cases where the best-known solution could not be attained are bcsstk29 and vibrobox. The number of edges for these two instances and bcsstk33 is at least twice than for the rest. This reduced hundreds of times the number of generations evolved by our algorithm, meaning that in these cases the way of managing the diversity of MAHMBCDP might not be adequate because it usually requires a lot of generations to reach high-quality solutions.

For the larger values of $k$, the performance of MAHMBCDP also degraded. Again, we could identify that the number of generations evolved was drastically reduced with the increase of $k$. In fact, in many cases, less than 1,000 generations could be evolved. Thus, the MAHMBCDP is really useful for those cases where many generations can be evolved. However, taking into account that several computationally expensive operators are used, the proposal is not scalable to large graphs (greater than about 17,000 edges) or to large values of $k$. In any case, for values of $k$ larger than 8, MAHMBCDP could identify five new best-known solutions.

As a summary, it is quite important to remark that best-known solution could be improved further in ten cases. Taking into account that these graphs have been tackled with a very large amount of proposals, this achievement is an important proof of the proper performance of the solver designed in this paper.

6 Conclusions and Future Work

The Graph Partitioning Problem (GPP) is a well-known NP-hard combinatorial optimization problem that has been addressed with numerous optimizers. Among them, hybrid metaheuristics that combine population-based approaches with trajectory-based schemes have reported the most promising results. Recently, the state-of-the-art of some related problems could be advanced significantly by taking into account the diversity of the population in an explicit way. This paper studies the applicability of these kinds of approaches in the GPP. Specifically, a memetic algorithm that incorporates a hill-climber and a relatively large neighborhood has been designed. The proposal (MAHMBCDP) incorporates a novel replacement strategy, the best-non-penalized scheme (BNP), that is based on considering both the diversity and quality in the survivor selection phase. Additionally, an important novelty is the design of a crossover operator that is based on the principle
Table 4. Summary of the results obtained with different \( \text{distInitFactor} \) values

<table>
<thead>
<tr>
<th>Instance</th>
<th>0.0</th>
<th>0.1</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>uk, ( k = 4 )</td>
<td>193.83</td>
<td>135</td>
<td>132.94</td>
<td>107.33</td>
<td>41</td>
<td>45.36</td>
<td>41</td>
</tr>
<tr>
<td>uk, ( k = 64 )</td>
<td>538.2</td>
<td>492</td>
<td>449.83</td>
<td>392</td>
<td>297.2</td>
<td>241</td>
<td>252.26</td>
</tr>
<tr>
<td>self, ( k = 4 )</td>
<td>375.9</td>
<td>270</td>
<td>201</td>
<td>201</td>
<td>201</td>
<td>201</td>
<td>201</td>
</tr>
<tr>
<td>self, ( k = 64 )</td>
<td>1558.23</td>
<td>1544</td>
<td>1538.56</td>
<td>1534</td>
<td>1537.4</td>
<td>1535</td>
<td>1537.13</td>
</tr>
</tbody>
</table>

Table 5. MAHMBCDP results for \( k = 4, 8 \)

<table>
<thead>
<tr>
<th>Instance</th>
<th>( k = 4 )</th>
<th>( k = 8 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TQPA</td>
<td>Best</td>
</tr>
<tr>
<td>add 20</td>
<td>1154</td>
<td>1151</td>
</tr>
<tr>
<td>data</td>
<td>382</td>
<td>382</td>
</tr>
<tr>
<td>self</td>
<td>201</td>
<td>201</td>
</tr>
<tr>
<td>uk</td>
<td>41</td>
<td>41</td>
</tr>
<tr>
<td>add 32</td>
<td>485</td>
<td>485</td>
</tr>
<tr>
<td>bcsstk3</td>
<td>21717</td>
<td>21717</td>
</tr>
<tr>
<td>whitaker3</td>
<td>381</td>
<td>381</td>
</tr>
<tr>
<td>crack</td>
<td>366</td>
<td>366</td>
</tr>
<tr>
<td>wing nodal</td>
<td>3575</td>
<td>3575</td>
</tr>
<tr>
<td>fe 3elt2</td>
<td>349</td>
<td>349</td>
</tr>
<tr>
<td>vibrobox</td>
<td>18976</td>
<td>19988</td>
</tr>
<tr>
<td>bcsstk29</td>
<td>8035</td>
<td>8069</td>
</tr>
<tr>
<td>4elt</td>
<td>326</td>
<td>326</td>
</tr>
<tr>
<td>cli</td>
<td>954</td>
<td>954</td>
</tr>
</tbody>
</table>

Table 6. MAHMBCDP results for \( k = 64 \)

<table>
<thead>
<tr>
<th>Instance</th>
<th>( k = 64 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TQPA</td>
</tr>
<tr>
<td>add 20</td>
<td>2949</td>
</tr>
<tr>
<td>data</td>
<td>2839</td>
</tr>
<tr>
<td>self</td>
<td>1532</td>
</tr>
<tr>
<td>uk</td>
<td>408</td>
</tr>
<tr>
<td>add 32</td>
<td>451</td>
</tr>
<tr>
<td>bcsstk3</td>
<td>107815</td>
</tr>
<tr>
<td>whitaker3</td>
<td>2491</td>
</tr>
<tr>
<td>crack</td>
<td>2535</td>
</tr>
<tr>
<td>wing nodal</td>
<td>15775</td>
</tr>
<tr>
<td>fe 3elt2</td>
<td>2478</td>
</tr>
<tr>
<td>vibrobox</td>
<td>4657</td>
</tr>
<tr>
<td>bcsstk29</td>
<td>55241</td>
</tr>
<tr>
<td>4elt</td>
<td>2565</td>
</tr>
<tr>
<td>3elt</td>
<td>3543</td>
</tr>
<tr>
<td>cli</td>
<td>5629</td>
</tr>
</tbody>
</table>

of maximizing the number of nodes that share the same class both in the parents and offspring.

The experimental validation has been performed with test cases of the Chris Walshaw's Graph Partitioning Archive. Comparisons against other strategies that were devised with the aim of facing premature convergence show the important benefits of the BNP strategy. It is also remarkable that in ten test cases, new best-known solutions could be attained. Taking into account the large number of techniques that have been used to deal with the GPP, this is a quite important achievement. The main drawback of the proposal is that due to the incorporation of computationally expensive components, there are issues with the scalability. Both the increase on the number of edges and the increase on the number of classes of the desired partitions provoke degradation on the performance.

Several lines of future work might be explored. First, some actions to improve the scalability might be studied. Particularly, taking into account more simple neighborhood definitions but with more complex trajectory-based schemes to intensify, seems a promising approach. Second, applying adaptive ways to manage the diversity might bring additional benefits. Finally, parallelizing the

doi: 10.13053/CyS-22-2-2951
Table 7. MAHMBCDP results for k = 16, 32

<table>
<thead>
<tr>
<th>Instance</th>
<th>TGPA</th>
<th>Best</th>
<th>Mean</th>
<th>Worst</th>
<th>TGPA</th>
<th>Best</th>
<th>Mean</th>
<th>Worst</th>
</tr>
</thead>
<tbody>
<tr>
<td>add20</td>
<td>2047</td>
<td>2041</td>
<td>2047.03</td>
<td>152</td>
<td>2362</td>
<td>2360</td>
<td>2362.8</td>
<td>2367</td>
</tr>
<tr>
<td>data</td>
<td>1127</td>
<td>1127</td>
<td>1127.8</td>
<td>122</td>
<td>1799</td>
<td>1799</td>
<td>1803.2</td>
<td>1813</td>
</tr>
<tr>
<td>3elt</td>
<td>573</td>
<td>573</td>
<td>573</td>
<td>573</td>
<td>960</td>
<td>960</td>
<td>961.53</td>
<td>962</td>
</tr>
<tr>
<td>uk</td>
<td>146</td>
<td>148</td>
<td>153.63</td>
<td>161</td>
<td>254</td>
<td>253</td>
<td>259</td>
<td>264</td>
</tr>
<tr>
<td>add32</td>
<td>2047</td>
<td>2041</td>
<td>2047.03</td>
<td>152</td>
<td>2362</td>
<td>2360</td>
<td>2362.8</td>
<td>2367</td>
</tr>
<tr>
<td>data</td>
<td>1127</td>
<td>1127</td>
<td>1127.8</td>
<td>122</td>
<td>1799</td>
<td>1799</td>
<td>1803.2</td>
<td>1813</td>
</tr>
<tr>
<td>3elt</td>
<td>573</td>
<td>573</td>
<td>573</td>
<td>573</td>
<td>960</td>
<td>960</td>
<td>961.53</td>
<td>962</td>
</tr>
<tr>
<td>uk</td>
<td>146</td>
<td>148</td>
<td>153.63</td>
<td>161</td>
<td>254</td>
<td>253</td>
<td>259</td>
<td>264</td>
</tr>
<tr>
<td>bcsstk33</td>
<td>54680</td>
<td>54687</td>
<td>54721.27</td>
<td>54800</td>
<td>77414</td>
<td>77508</td>
<td>77616.93</td>
<td>77824</td>
</tr>
<tr>
<td>whitaker</td>
<td>1088</td>
<td>1088</td>
<td>1089.47</td>
<td>1090</td>
<td>1614</td>
<td>1619</td>
<td>1628.67</td>
<td>1642</td>
</tr>
<tr>
<td>crack</td>
<td>1088</td>
<td>1088</td>
<td>1089.47</td>
<td>1090</td>
<td>1614</td>
<td>1619</td>
<td>1628.67</td>
<td>1642</td>
</tr>
<tr>
<td>wing_nodal</td>
<td>8334</td>
<td>8333</td>
<td>8341.77</td>
<td>8377</td>
<td>11768</td>
<td>11774</td>
<td>11793.13</td>
<td>11825</td>
</tr>
<tr>
<td>te_delt2</td>
<td>1007</td>
<td>1008</td>
<td>1008.77</td>
<td>1010</td>
<td>1614</td>
<td>1619</td>
<td>1628.67</td>
<td>1642</td>
</tr>
<tr>
<td>vibrbox</td>
<td>31892</td>
<td>32428</td>
<td>32744.6</td>
<td>32920</td>
<td>39477</td>
<td>39443</td>
<td>39699.7</td>
<td>39975</td>
</tr>
<tr>
<td>4elt</td>
<td>934</td>
<td>937</td>
<td>957.13</td>
<td>1084</td>
<td>1551</td>
<td>1554</td>
<td>1566.9</td>
<td>1597</td>
</tr>
<tr>
<td>te_sphere</td>
<td>1714</td>
<td>1719</td>
<td>1875.63</td>
<td>2068</td>
<td>2490</td>
<td>2500</td>
<td>2561.3</td>
<td>2659</td>
</tr>
<tr>
<td>cti</td>
<td>2793</td>
<td>2799</td>
<td>2846.37</td>
<td>2895</td>
<td>4046</td>
<td>4069</td>
<td>4114.2</td>
<td>4153</td>
</tr>
</tbody>
</table>

Proposal to reduce the time required to obtain high-quality solutions seems a plausible approach.

Acknowledgements

The authors acknowledge funding from Conacyt project no. 285599 "Toma de decisiones multiobjetivo para sistemas altamente complejos".

References


Article received on 01/02/2017; accepted on 16/05/2017. Corresponding author is Emmanuel Romero Ruiz.