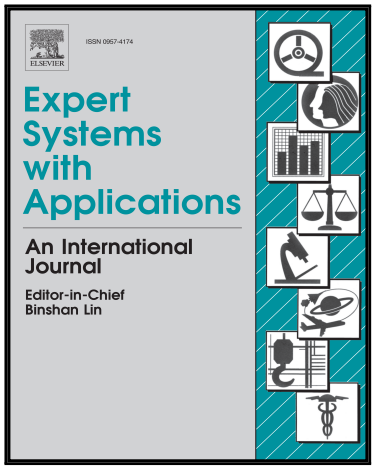
Accepted Manuscript



Multi-objective Memetic Meta-heuristic Algorithm for Encoding the Same Protein with Multiple Genes

Belen Gonzalez-Sanchez, Miguel A. Vega-Rodríguez, Sergio Santander-Jiménez

PII: S0957-4174(19)30431-2
DOI: https://doi.org/10.1016/j.eswa.2019.06.031
Reference: ESWA 12743

To appear in: *Expert Systems With Applications*

Received date: 11 October 2018
Revised date: 10 May 2019
Accepted date: 15 June 2019

Please cite this article as: Belen Gonzalez-Sanchez, Miguel A. Vega-Rodríguez, Sergio Santander-Jiménez, Multi-objective Memetic Meta-heuristic Algorithm for Encoding the Same Protein with Multiple Genes, *Expert Systems With Applications* (2019), doi: https://doi.org/10.1016/j.eswa.2019.06.031

Highlights

- The design of multiple genes encoding the same protein is an important task
- We have re-defined this multi-objective problem from three to two objectives
- We have designed and implemented a solution procedure based on the MOSFLA algorithm
- The experiments have been done over 9 real protein instances
- MOSFLA-2CH obtains better results than the ones found in the literature

Multi-objective Memetic Meta-heuristic Algorithm for Encoding the Same Protein with Multiple Genes

Belen Gonzalez-Sanchez[a], Miguel A. Vega-Rodríguez[b,*], Sergio Santander-Jiménez[c]

[a]*Escuela Politécnica, Universidad de Extremadura, Avda. de la Universidad s/n, 10003 Cáceres, Spain.*
[b]*Instituto de Investigación en Tecnologías Informáticas Aplicadas de Extremadura (INTIA), Universidad de Extremadura, Avda. de la Universidad s/n, 10003 Cáceres, Spain.*
[c]*INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, 1000-029 Lisboa, Portugal.*

Abstract

An important goal in synthetic biology is to maximize the expression levels of proteins. For this purpose, multiple genes encoding the same protein can be integrated into the host genome. However, this approach is affected by two key issues. Firstly, codons with better adaptation indexes should be used, since some synonymous codons are better adapted than others. Secondly, the multiple protein-coding sequences should be as different as possible to avoid the loss of gene copies due to homologous recombination. Therefore, this task shows strict biological requirements that make it difficult to tackle. In this work, we design and implement a computational intelligence approach to address this problem, the Multi-Objective Shuffled Frog Leaping Algorithm (MOSFLA). This method combines the optimization capabilities provided by parallel searches, multiple operators, and memetic strategies to tackle problems with difficult solution quality requirements. Several alternatives have been comparatively analyzed, including MOSFLA variants with three objectives as in other approaches from the literature and also variants with only two objectives. Experiments on nine real-world protein datasets give account of the improved, statistically significant

*Corresponding author
Email addresses: belengs@unex.es (Belen Gonzalez-Sanchez), mavega@unex.es (Miguel A. Vega-Rodríguez), sergio.jimenez@tecnico.ulisboa.pt (Sergio Santander-Jiménez)

performance achieved over the related work, attending to different quality metrics, confirming that our proposal satisfactorily deals with the complex nature of the problem.

Keywords: Multi-objective Memetic Meta-heuristic Algorithm, Design of Multiple Genes, Encoding of the Same Protein, Multi-objective Optimization, Protein-Coding Sequence (CDS).

1. Introduction

To maximize the expression levels of a protein is a major goal in synthetic biology. An encouraging and widely-used strategy to attain this is the integration of multiple genes that encode the same protein. In essence, with this strategy, it is expected that the expression levels increase proportionally to the number of copies of the gene (e.g. see Vassileva et al. (2001)), but this does not happen in each and every case (Hohenblum et al. (2004)). Even so, it is a commonly used strategy in the last years to optimize the expression levels of a protein, for instance, in Gu et al. (2015); Tyo et al. (2009); Scorer et al. (1994).

However, this is not a simple task and the integration of multiple gene copies implies a time-consuming process with complex biological requirements. Some related works (Gu et al. (2015); Tyo et al. (2009); Scorer et al. (1994)) propose to simplify it by integrating the copies very near each other within the organism genome. But this approach entails an important inconvenient, the homologous recombination. For example, if there are six copies of a gene (g_1, g_2, g_3, g_4, g_5, g_6) sequentially concatenated and a homologous recombination occurs between g_2 and g_5, the genes g_3 and g_4 are lost as a result. This means that, when identical sequences are very close, they can induce homologous recombination and consequently some sequences could be lost (Aw & Polizzi (2013)).

Following that argument, and taking into account those issues, the need to make each protein-coding sequence (named CDS) as different as possible arises. In particular, a CDS must be different with regard to other CDSs and with the subsequences that represent the CDS itself. The minimum length of a sequence

that induces homologous recombination is not fixed and it depends on the host organism like some authors reported. For example, Shen & Huang (1986) set in 23 bp (base pairs) the minimum length of subsequence to induce homologous recombination in *Escherichia coli*. In *Bacillus subtilis*, Khasanov et al. (1992) report that homologous recombination is caused by identical sequences with 70 bp as length. Other example, Manivasakam et al. (1995) describe that, in *Saccharomyces cerevisiae*, identical sequences greater than 30 bp increase highly the likelihood of homologous recombination. In general, the minimum length that induces homologous recombination is not known, but all studies agree that it is important to minimize the length of identical sequences in order to reduce the homologous recombination rate.

In order to make the CDSs of the same protein different among them, each amino acid can be encoded with different synonymous codons, so we can get different CDSs for the same protein. Although the synonymous codons represent the same amino acid, some synonymous codons are more frequent in some organisms and selecting a synonymous codon or another can affect the expression levels of a protein (Athey et al. (2017)). As a consequence, it is relevant which synonymous codon is selected, being the best option the synonymous codons with the highest usage frequencies, that is, the most highly adapted ones.

Hence, protein encoding represents a difficult task in which potential solutions must meet multiple, strict biological properties. Three main objectives can be identified: 1) to minimize the length of the longest repeated sequence within the same CDS; 2) to maximize the differences between two CDSs; and 3) to maximize the adaptation to the host organism, given by the Codon Adaptation Index (CAI). The hard-to-tackle nature of this multi-objective problem requires the proposal of robust approaches based on computational intelligence principles to handle such solution quality requirements. Accordingly, this paper is aimed at addressing protein encoding by designing and implementing a multi-objective approach, named Multi-Objective Shuffled Frog Leaping Algorithm (MOSFLA), which combines different algorithmic learning strategies, memetic searches, and knowledge sharing to tackle complex optimization problems.

MOSFLA contributes with an advanced algorithmic design that incorporates multiple strategies for improved optimization accuracy. It is based on the idea of evolving separately different partitions of solutions, known as memeplexes. Each memeplex is subject to processing under both standard operators and expert, problem-oriented mutations, with the aim of generating new candidate solutions that meet the quality stipulations of the tackled problem. In addition, the algorithm dynamically exploits the best local / global solutions according to the multi-objective status of the optimization process, including re-initialization mechanisms to deal with potential stagnation issues. Finally, a shuffling procedure is used to distribute solutions among memeplexes at each generation, thus allowing a global exchange of knowledge from the conducted local searches.

The relevance and utility of the proposal then lies in the advanced optimization capabilities provided by the combined search mechanisms within MOSFLA design, including dynamic exploitations/explorations through if-else rules, concurrent search space processing, and solutions shuffling for knowledge spreading. Through experimentation on nine real-world problem instances, we show how the solution of complex problems governed by specific quality requirements (as in the case of multi-objective protein encoding) can benefit from the joint action of the implemented intelligent algorithmic strategies.

Therefore, an important contribution of this work is the proposal of an expert and intelligent approach, MOSFLA, to solve this complex optimization problem from the synthetic biology field. Different works can be found in the current literature explaining how to improve, from a biological point of view, the expression levels of proteins in different organisms, but neither of them proposes a computational approach as MOSFLA, based on the joint action of different intelligent algorithmic strategies. More specifically, Song et al. (2017) focused on *Arachis duranensis* and *Arachis ipaënsis* (gene sources for research in plant biology of peanut). They showed that highly expressed coding sequences had higher codon adaptation. Chen et al. (2017) applied their study to the yeast *Saccharomyces cerevisiae*. They stated that codon adaptation is stronger in more highly expressed genes, a phenomenon commonly explained by stronger natural

selection on translational accuracy and/or efficiency among these genes. Their study revealed the pleiotropic effects of synonymous codon usage and provided an additional explanation for the correlation between codon adaptation and gene expression level. Vasanthi & Dass (2018) analyzed different bacterial species (*Pseudomonas fuscovaginae*, *Pseudomonas syringae*, *Xanthomonas oryzae*, and *Pseudomonas avenae*) infecting rice. They concluded that certain genes with high CAI have been correlated for better gene expression. Zhang et al. (2018) focused on 12 *Solanum* species, which is one of the largest genera, including two important crops - potato (*Solanum tuberosum*) and tomato (*Solanum lycopersicum*). They compared the chloroplast codon usage bias among the 12 species, between photosynthesis-related genes (Photo-genes) and genetic system-related genes (Genet-genes). Among their findings, they obtained that Photo-genes had higher codon adaptation indexes than Genet-genes, indicative of a higher gene expression level and a stronger adaptation of Photo-genes. Sahoo et al. (2019) applied their study to *Arabidopsis thaliana* (a weed found in roadsides and disturbed lands). They found a systematic strong correlation between CAI and gene expression measures. Finally, Wang et al. (2019) studied the Newcastle disease virus (NDV, a contagious viral bird disease affecting many domestic and wild avian species, which is transmissible to humans). They used a codon modification strategy to attenuate the three major virulence factors: the fusion (F) protein, hemagglutinin neuraminidase (HN), and phosphoprotein (P). Recoding the F and HN genes with rare codons (codons with low CAI) greatly reduced expression of both F and HN proteins and resulted in their low incorporation into virions. Moreover, full attenuation was achieved when the P gene was recoded.

Continuing with the related work, several recent studies (Webster et al. (2017); Yu et al. (2015); Tran et al. (2015)) analyzed the codon usage frequency optimization in the host organism and other studies offer tools, for example COOL (Chin et al. (2014)), D-Tailor (Guimaraes et al. (2014)) or OPTIMIZER (Puigbò et al. (2007)) for the same purpose. However, our approach differs from these previous studies in that we also optimize the differentiation between the

CDSs that encode the same protein, in order to avoid the homologous recombination (a very negative effect). Following with the literature review, a recent work (Terai et al. (2017)) has been published with the same purpose as this paper. Therefore, we perform comparisons with the results from this previous work. Our approach uses a different algorithm, Multi-Objective Shuffled Frog Leaping Algorithm (MOSFLA), and as we will see, we have introduced improvements in the definition of the problem. On the one hand, our multi-objective proposal for protein encoding is inspired by the baseline skeleton of the Shuffled Frog Leaping Algorithm (SFLA, Elbeltagi et al. (2007)), an approach based on the natural behavior of frogs. We have focused on SFLA because different examples of successful applications (such as water distribution and power flow optimization, production planning, project management, multi-user detection, etc.) point out the relevance of SFLA to address complex problems (Sarkheyli et al. (2015)). In fact, SFLA has led to good results in other difficult bioinformatics problems, including RNA secondary structure prediction (Lin et al. (2012)) and biomedical data feature selection (Hu et al. (2018)).

On the other hand, after the first experiments, we deduced that two of the three objectives are correlated. That is, the improvement in one of these two objectives also produces the improvement in the other objective. So, we have improved and simplified the problem definition by using only two objectives (instead of three objectives). As we will see, after a comparative study with three different quality metrics and the corresponding statistical analyses, we can conclude that our approach obtains very good results in comparison with the related work, with statistically significant improvements.

Hence, the main contributions of this work can be summarized as:

- Proposal of a multi-objective approach based on the memetic metaheuristic SFLA, adapting it (several mutation operators, greedy initialization for one of the solutions, etc.) to the design of multiple genes encoding the same protein.

- Identification of possible correlated objectives in the optimization prob-

lem, comparing the results when three and different combinations of two objective functions are used. This study leads to the conclusion that the tri-objective optimization problem could be turned into a bi-objective optimization problem.

- Evaluation of the proposal over nine real-world protein datasets, undertaking a thorough statistical analysis based on the results from three important quality indicators.

- Comparison with the state-of-the-art multi-objective approach (Terai's method, proposed in 2017, Terai et al. (2017)) for solving this multi-objective optimization problem, surpassing the best results found in the literature, with improvements that are statistically significant.

Furthermore, if we focus on other previous expert and intelligent systems based on SFLA, such as Niknam et al. (2011), Luo & Chen (2014), Luo et al. (2014), Zhu & Zhang (2014), and Lei & Guo (2015), the following theoretical contributions can be added to the previous ones:

- We have incorporated a multi-objective algorithmic design to SFLA, including multi-objective, quality-oriented population partitioning to perform multiple, parallel searches over different sets of solutions.

- This multi-objective algorithmic design implies the comparison of solutions based on the dominance operator, the use of non-dominated sortings, and the application of the crowding multi-objective metric (a diversity metric).

- MOSFLA includes both standard operators and expert, problem-aware mutations (a total of four operators have been added into its algorithmic design) to effectively deal with the different problem objectives.

The rest of this paper is organized as follows. Section 2 explains and gives a formal definition of the multi-objective optimization problem to solve. After that, Section 3 details and describes our approach (MOSFLA) to tackle this

optimization problem. Section 4 includes the experimental settings, the experiments performed, the re-definition of the problem, the results obtained, the comparisons with the results found in the literature, and the corresponding statistical analyses. Finally, Section 5 explains the conclusions of this work and indicates possible future lines.

2. Problem Definition

In this multi-objective optimization problem, a solution represents an encoded protein. Each solution is composed of a set of sequences (CDSs) of equal length. Also, the user sets the number of CDSs for an encoded protein. Each CDS encodes the sequence of amino acids that defines the protein using the set of the synonymous codons. The final representation of a CDS is a string of characters. An example of encoded protein is shown in Figure 1.

In order to optimize the solutions, each one is assessed by three objective functions. The first objective is related to the encoding of each amino acid using synonymous codons (the preferred option is the codon with a higher usage frequency). The second and third objective functions are based on avoiding identical subsequences between two CDSs and reducing repeated subsequences in a CDS itself, respectively. These three objective functions are detailed in each one of the following subsections.

2.1. Codon Adaptation Index (CAI)

The aim of the first objective is to maximize the minimum Codon Adaptation Index ($mCAI$) value of a solution. Each sequence (CDS) from a solution has a CAI value which is dependent on the synonymous codons used to encode the protein. Some codons are better adapted than others so they have higher usage frequency value than others. For this objective, the best selection would be to use the codons with the highest frequency value. Equation 1 calculates this objective function.

$$mCAI = \min_{1 \leq i \leq I} \quad CAI(CDS_i), \tag{1}$$

where CDS_i is each CDS that encodes the protein, I is the number of CDSs, and CAI value is calculated for each one as indicated by Equation 2.

$$CAI(CDS_i) = \sqrt[N]{\prod_{n=1}^{N} W(codon_{i,n})}, \tag{2}$$

where N is the number of codons that CDS_i has and W is the weight assigned to the $codon_{i,n}$. This weight is calculated as the usage frequency of $codon_{i,n}$ relative to (divided by) the usage frequency of the most frequent codon among the synonymous codons of $codon_{i,n}$ (Sharp & Li (1987)). The usage frequencies have been obtained from the research carried out by Terai et al. (2017).

This objective is focused on optimizing the minimum CAI value, since the average of I CDSs is not representative. For example, it could happen that the i-th CDS has a very low CAI value within a good CAI average. For this reason, we have not used the average. Therefore, this objective function maximizes the minimum CAI ($mCAI$) value, trying that all the CDSs have so high CAI values as possible.

2.2. Hamming Distance between CDSs (HD)

The second objective function seeks the pair of CDSs which contains more identical subsequences (same subsequences in the same positions). The objective function calculates a measure based on the normalized Hamming Distance (HD) between two CDSs. In particular, the objective is to compute the Hamming distance value between all possible pairs combinations and focuses on maximizing the minimum Hamming Distance (mHD) value as shown in Equation 3.

$$mHD = \min_{1 \le i < j \le I} \frac{HD(CDS_i, CDS_j)}{L}. \tag{3}$$

For a pair of CDSs, CDS_i and CDS_j, both with length L nucleotides, the Hamming distance is calculated as indicated by Equation 4.

$$HD(CDS_i, CDS_j) = \sum_{1 \le k \le L} \sigma(CDS_{i,k}, CDS_{j,k}), \tag{4}$$

where the i-th and j-th CDSs are compared and, in both CDSs, the k-th nucleotide is evaluated. If $CDS_{i,k}$ and $CDS_{j,k}$ are equal then σ is 0. However, if they are different nucleotides, σ is set to 1.

As in the case of the first objective function, a very low HD value could be unnoticed within a good average. Thus, the objective function is to maximize the minimum value instead of the average value.

2.3. Length of Repeated or Common Substrings (LRCS)

The third objective function is based on breaking repeated subsequences occurring between a pair of CDSs or within the same CDS. The objective is to decrease the longest length of repeated or common substring (LRCS).

We say that we find a common substring $S_{i,p,l}$ in the i-th CDS, at the p-th position with a length of l characters (nucleotides), when the same or another CDS (j-th CDS) has the same substring $S_{j,q,l}$ at the same (in this particular case, $i \neq j$) or different q-th position.

For example, in Figure 1, $AGCGUUU$ is the longest common substring between all pairs of CDSs, although there are other repeated substrings, e.g. in CDS_1, $GAGA$, that have a shorter length. The objective is to minimize the maximum length of the repeated or common substrings ($MLRCS$) found as defined in Equation 5.

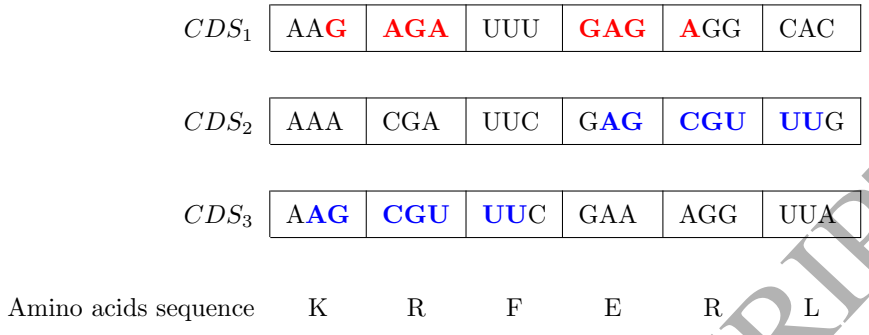


Figure 1: A possible solution with 3 CDSs and 18 nucleotides per CDS that shows an example for the computation of the repeated or common substrings. $AGCGUUU$ is the longest common substring, although there are other repeated substrings, e.g. in the same CDS (CDS_1), $GAGA$, but this one has a shorter length.

$$MLRCS = \max_{1 \leq i \leq j \leq I} \frac{LRCS(CDS_i, CDS_j)}{L}, \tag{5}$$

where L is the length in nucleotides of the CDSs and $LRCS$ is calculated for every couple of CDSs, CDS_i and CDS_j, enabling $i = j$ in order to seek into the same CDS, as shown in Equation 6.

$$LRCS(CDS_i, CDS_j) = \max_{1 \leq p,q,l \leq L} length(S_{i,p,l}) \quad when\ (S_{i,p,l} = S_{j,q,l}), \tag{6}$$

where, again, L is the length of the CDSs, and if $p = q$ then $i \neq j$.

3. Multi-Objective Shuffled Frog Leaping Algorithm (MOSFLA)

The Shuffled Frog Leaping Algorithm (SFLA, Elbeltagi et al. (2007)) is a memetic meta-heuristic based on the behavior of frogs and the evolution of groups (memeplexes). SFLA is designed to solve optimization problems by performing local searches and global information exchange.

SFLA starts by creating an initial frog population (set of solutions) and then defines its main operations, which are executed iteratively to improve and mix

the population, so as to change the relative positions inside of a memeplex or to exchange individuals (solutions) among memeplexes.

After the initialization, the individuals are sorted by their individual fitness values, which have been calculated previously, and the algorithm continues with the shuffle of the population. The individual with the best fitness is located in the first memeplex, the second best individual is located in the second memeplex and so on, until all memeplexes have an individual. Then, the algorithm assigns the next individual to the first memeplex, the following one to the second memeplex and so on, until there are no more individuals to distribute.

Once the population has been divided into memeplexes, each memeplex tries to improve its individuals locally during several attempts. Finally, all memeplexes are joined and all individuals are re-sorted by using their new fitness values. This sorting, dividing, improving, and joining population process is repeated multiple times until the stop criterion is met.

In our approach, we focus on a multi-objective (MO) optimization problem, so we have adapted SFLA to the multi-objective context and to the particular problem under study (therefore, designing and implementing the algorithm MOSFLA). In the multi-objective context, we do not have a unique best solution, but a set of trade-off solutions that optimize at least one of the objectives. Therefore, we use the dominance concept to compare solutions. We say that a solution x dominates another solution y ($x \succ y$), or that y is dominated by x, when x gets a better or equal value than y for each objective function and at least one of them is better. On the other hand, a solution x is non-dominated or Pareto optimal if there is not any solution that dominates it. Finally, the set of non-dominated solutions is known as Pareto set and its graphical representation as Pareto front. This Pareto set is the output of a multi-objective optimization algorithm, and it is used for applying quality metrics and comparing results.

Algorithm 1 shows the pseudo-code of the proposed MOSFLA. At the beginning, a file for storing the non-dominated solutions is created. Also, the population is initialized with *population_size* individuals or solutions (line 2). Each individual is randomly generated, except one which is generated by se-

lecting the codon with the best CAI for each amino acid so the value of $mCAI$ objective function is 1 for this greedy solution. This specific solution is always non-dominated and it is created as possible aid to achieve high values of $mCAI$ in the next generations.

Before the cycle loop starts, the population is evaluated and sorted by using two multi-objective metrics: rank and crowding (line 3). The first one indicates in which layer of the generated Pareto fronts a solution belongs to and it is based on the dominance relationships among all the solutions. The second one estimates the density of the solutions, preferring solutions with higher crowding distances, that is, more diverse solutions. More details about these two multi-objective metrics ca be found in Deb et al. (2002). In conclusion, the population is sorted by quality in order to be shuffled in the different memeplexes.

Algorithm 1 MOSFLA pseudo-code.

Input: *population_size* (number of solutions), m (number of memeplexes), *max_cycles* (maximum number of generations), P_m (mutation probability)

Output: *nondominated_file* (set of non-dominated solutions saved in a file)

1: $nondominated\_file \leftarrow \emptyset$;
2: $population \leftarrow init\_population(population\_size)$;
3: $population \leftarrow order\_by\_R\&C(population, population\_size)$;
4: **for** $cycle \leftarrow 1, max\_cycles$ **do**
5: $memeplexes \leftarrow divide\_population(population, population\_size, m)$;
6: $memeplexes \leftarrow improve\_memeplexes(memeplexes, m, P_m)$;
7: $population \leftarrow merge\_order\_by\_R\&C(memeplexes, 2 * population\_size)$;
8: $nondominated\_file \leftarrow save\_nondominated\_solutions(population)$;
9: **end for**

Next, the for loop starts. It includes operations that make the frog population evolve for *max_cycles* cycles or generations. Each cycle involves the management of the population and the memeplexes.

In the first step (line 5), the sorted population is divided into m memeplexes, and shuffled as explained before. Therefore, the *population_size* must be mul-

tiple of the number of memeplexes (m) and every memeplex has to have equal number of individuals.

The following step (line 6) tries to improve, in a local way, each memeplex. It is shown in detail in Algorithm 2. In this algorithm, the global best solution (in the whole population) is identified by X_B (Algorithm 2, line 1), while the local best and worst solutions within each memeplex are identified by X_b and X_w, respectively. Here, the memeplexes are iteratively processed to improve the worst local solution by applying a mutation operator (Algorithm 2, lines 2-3). This operator executes one of four possible types of mutation in a random way. That is, one of them is selected and the four have the same probability to be selected. Furthermore, once selected the mutation type, all of them use a probability of P_m. Every kind of mutation consists in changing a specific part of a solution:

1. Changing each codon of each CDS by another random synonymous codon with a probability of P_m.
2. Changing each codon of the CDS with the lowest CAI value by another synonymous codon with a probability of P_m. The new codon must have higher weight (usage frequency) than the replaced codon, otherwise, the codon is not changed.
3. Changing each codon of the pair of CDSs with the lowest Hamming distance by another random synonymous codon with a probability of P_m.
4. Changing each codon belonging to the longest common substring, in the same CDS or in different CDSs, by a random synonymous codon with a probability of P_m.

Related to the X_w improvement, firstly, a new solution (X_n) is created from a mutated X_b (Algorithm 2, line 6), that is, considering the local best solution. If X_n does not improve (dominates X_w) then X_n is created from a mutated X_B (Algorithm 2, line 10), that is, considering the global best solution. Lastly, if X_n does not improve (dominates X_w) then X_n is created as a new random solution (Algorithm 2, line 14).

Algorithm 2 *improve_memeplexes*$(memeplexes, m, P_m)$ pseudo-code.

Input: *memeplexes* (all the memeplexes), m (number of memeplexes), P_m (mutation probability), *max_improv* (number of improvements per memeplex)

Output: *memeplexes* (Set of solutions of size $2 * population\_size$)

```
1:  X_B ← Select_global_best_solution(memeplexes);
2:  for i ← 1, m do
3:    for improv ← 1, max_improv do
4:      X_b, X_w ← Select_local_best_and_worst_solution(memeplexes, i);
5:      save_worst(memeplexes, X_w);
6:      X_n ← mutation(X_b, P_m);
7:      if X_n ≻ X_w then
8:        save_solution(memeplexes, X_n, position_X_w);
9:      else
10:       X_n ← mutation(X_B, P_m);
11:       if X_n ≻ X_w then
12:         save_solution(memeplexes, X_n, position_X_w);
13:       else
14:         X_n ← init_random_solution();
15:         save_solution(memeplexes, X_n, position_X_w);
16:       end if
17:     end if
18:     order_by_R&C(memeplexes, i);
19:   end for
20: end for
```

This improvement process generates a child population along with the parent population. In particular, when X_n is selected, it is stored in the position previously occupied by X_w (Algorithm 2, lines 8, 12, and 15). For this reason, the replaced solution X_w is previously saved in the backup population (Algorithm 2, line 5). As we have two populations, we can also say that, at the end of the improvement process, the population has a size equal to $2 * population\_size$.

Finally, each memeplex i reorders its individuals by rank and crowding in order to precisely select its next X_b and X_w solutions (Algorithm 2, line 18).

After finishing the improvements in all the memeplexes, Algorithm 1 continues. The full population with $2 * population\_size$ individuals is sorted by rank and crowding (Algorithm 1, line 7) in order to reduce the population to the half, that is, its original size, for the next cycle. Furthermore, the non-dominated solutions are stored in the $nondominate\_file$ (Algorithm 1, line 8).

After explaining the MOSFLA pseudo-code, we can highlight some of its advantages. It is a multi-objective memetic meta-heuristic algorithm based on a cooperative population of frogs (swarm intelligence). Thus, it merges local searches, and global exchange and shuffle periodically. Also, it provides the chance to include new individuals generated randomly when a solution is not improved.

In this study, this algorithm has been implemented in C/C++. The next section includes the experimental settings (the data sets used in the experiments, the parameter configuration for MOSFLA, etc.), the experiments performed, the re-definition of the problem, the results obtained, the comparisons with the results found in the literature, and the corresponding statistical analyses.

4. Experiments and Results

4.1. Experimental settings

As mentioned in previous sections, we compare MOSFLA's results with the results from Terai et al. (2017), a previous work with the same purpose as this paper, that is, addressing the same problem. To make a fair comparison between algorithms, we have selected nine real proteins as a representative sample based on two attributes: length or number of amino acids (AA) and number of CDSs. These two attributes have a direct impact in the complexity of the instance, so the selected proteins cover a wide range of different situations. As we can observe in Table 1, we have chosen nine very different proteins in terms of length

and number of CDSs. We have used the Universal Protein Resource (UniProt[1]) to get the FASTA format for every protein.

Code	Name	CDSs	Length (AA)	CDSs*Length
Q5VZP5	DUS27_HUMAN	2	1158	2316
A4Y1B6	FADB_SHEPC	3	716	2148
B3LS90	OCA5_YEAS1	4	679	2716
B4TWR7	CAIT_SALSV	5	505	2525
Q91X51	GORS1_MOUSE	6	446	2676
Q89BP2	DAPE_BRADU	7	388	2716
A6L9J9	TRPF_PARD8	8	221	1768
Q88X33	Y1415_LACPL	9	114	1026
B7KHU9	PETG_CYAP7	10	38	380

Table 1: List of proteins used in the experiments.

To get the results from the multi-objective method implemented by Terai et al. (2017), we have used their web-based application[2] with its default settings (those proposed by its authors) for the nine instances. In next subsections, these results are compared, instance by instance, with the results obtained by our approach.

In our method, the parameter configuration is established so that we can make a fair comparison between both methods. In particular, the *population_size* is equal to 100 individuals (or solutions) and the number of generations (*max_cycles*) is set to 100. Both parameters are established with the same values for the two methods. Furthermore, both methods have used the same population initialization procedure. In addition, we tune the other parameters of MOSFLA: number of memeplexes (m) and mutation probability (P_m). As shown in Table 2, for these two parameters, different values have been tested in order to find the best

[1] http://www.uniprot.org/uniprot/

[2] http://tandem.trahed.jp/tandem/

configuration (highlighted in bold). Also, as we can observe in Algorithm 2, the number of improvements per memeplex (max_improv) needs to be established, but in this case, in order to make fair comparisons, this number is equal to the number of solutions per memeplex, that is, $population\_size/m$; thus performing only one improvement per individual.

Moreover, in all the following sections, we have executed every experiment 31 times (independent runs) in order to ensure reliable statistics due to the stochastic nature of MOSFLA.

	Checked values						
P_m	0.3125%	0.625%	**1.25**%	2.5%	5%	10%	20%
m	2	5	**10**	20			

Table 2: All tested values to find the best configuration. The best value is highlighted in bold.

After analyzing all the results from the experiments, we have set the nadir and ideal values for each objective as indicated in Table 3. These values are used in all the next sections. Taking into account these values for all the problem instances, all the objective functions are normalized in the range [0,1], so the graphical representations and the computation of the quality metrics are performed over normalized values.

Objective	**Nadir value**	**Ideal value**
mCAI	0	1
mHD	0	0.40
MLRCS	1	0

Table 3: Nadir and ideal values used in the computation of the quality metrics and normalizations for all the proteins.

4.2. First results with MOSFLA and 3 objective functions

We have observed in the first results that the objectives mHD and MLRCS are not in conflict with each other, that is, optimizing one objective does not

negatively affect the other objective. As an example, Figure 2 shows the median Pareto front obtained for one of the proteins, $Q88X33$, and we can confirm that solutions with good score for the mHD objective also have a good value for the MLRCS objective, and vice versa. That is, these two objectives have some correlation. By contrast, the first objective (mCAI) is essential and it is clearly in conflict with the other two objectives (mHD and MLRCS). In fact, if the mCAI objective improves then the mHD and MLRCS objectives are worse, and vice versa.

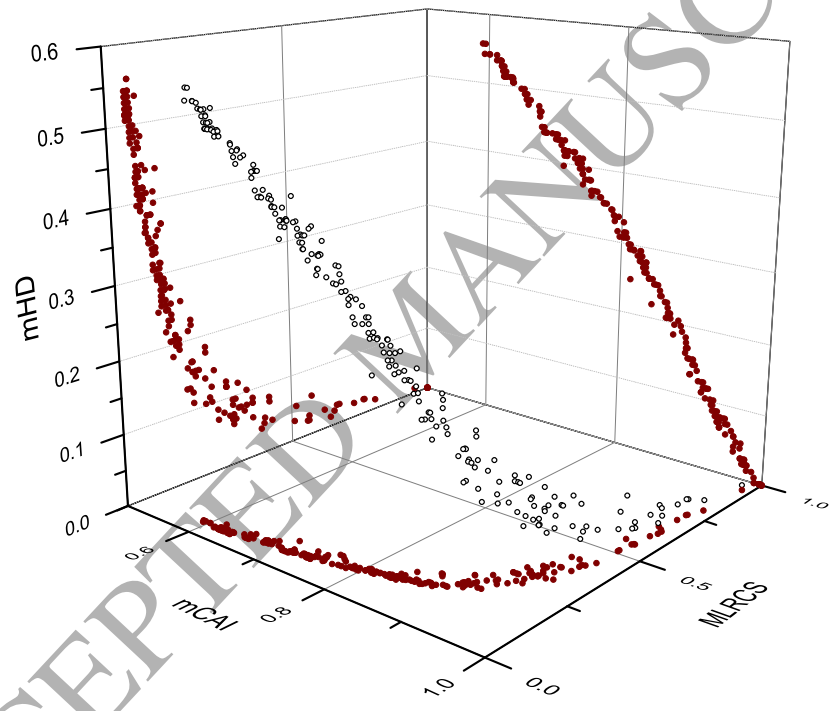


Figure 2: 3D scatter plot of the median Pareto front obtained by MOSFLA for the $Q88X33$ protein. The points in the different 2D projections appear in red.

For this reason, and in contrast to Terai et al. (2017), we propose a re-definition of the problem. We think it is better to use only two objectives instead of three objectives. More specifically, we propose to optimize the first

objective (mCAI) and one of the other two objectives. In the next subsection, we evaluate this proposal, comparing the results obtained by MOSFLA with 3 and 2 objective functions. In the case of 2 objective functions, both alternatives are evaluated: optimizing mCAI-mHD and optimizing mCAI-MLRCS.

4.3. Comparing the results when 3 and 2 objective functions are used

To evaluate and compare the quality of the results obtained by MOSFLA when 3 and 2 objective functions are optimized, we have adopted the quality metric maybe most widely-used in multi-objective optimization: the hypervolume (Beume et al. (2009)).

The hypervolume (HV) indicator, also known as Lebesgue measure (Bartle (1995)), is a well-established unitary indicator of a Pareto front's quality. In the case of three objectives, it measures the volume (in percentage) of the objective space portion dominated by a Pareto front $\mathcal{A}$. It is calculated as Equation 7 indicates.

$$HV(\mathcal{A}, r) = Leb\left(\bigcup_{i=1}^{|\mathcal{A}|} h(a_i, r)\right), \tag{7}$$

where Leb refers to the Lebesgue measure, $|\mathcal{A}|$ is the size (cardinality) of the set $\mathcal{A}$, and $h(a_i, r)$ is the volume defined by each point $(a_1, a_2, \ldots, a_{|\mathcal{A}|})$ and the reference point r.

In order to distinguish the three MOSFLA variants. The variant that optimizes the three objectives is called MOSFLA. The variant that optimizes the two objectives mCAI-mHD is named MOSFLA-2CH, while the variant that optimizes the two objectives mCAI-MLRCS is called MOSFLA-2CL. In any of these cases, the comparisons are made with three objectives. That is, the variants that only optimize two objectives are also compared by using the three objectives. Our goal is to know if it is possible to obtain better (or similar) results in the three objective space, even when only two of them are optimized. In conclusion, the HV indicator has been always calculated in the same way (taking into account the three objective space).

Table 4 shows the median HV results for each protein that were obtained by the three different variants of MOSFLA. We can observe that the results from MOSFLA-2CH are slightly better or similar than the results from MOSFLA (3 objectives) in almost all the instances, obtaining a very good average HV. The same cannot be said for MOSFLA-2CL. That is, if the second objective (mHD) is not optimized, we obtain worse Pareto fronts than if the third objective (MLRCS) is not optimized. In other words, we can say that when we optimize the mHD objective, we are also optimizing the MLRCS objective; generating very good results in the three objective space. For this reason, in the following sections, we use the variant MOSFLA-2CH (optimizing only the two objectives mCAI and mHD) as our best approach. This demonstrates that the re-definition of the problem is possible, using only two objectives (mCAI and mHD).

Protein	**MOSFLA**	**MOSFLA-2CH**	**MOSFLA-2CL**
Q5VZP5	$64.48\%_{\pm 0.72\%}$	$\mathbf{65.77\%}_{\pm 0.36\%}$	$40.83\%_{\pm 1.05\%}$
A4Y1B6	$53.56\%_{\pm 0.19\%}$	$\mathbf{53.81\%}_{\pm 0.25\%}$	$35.29\%_{\pm 2.62\%}$
B3LS90	$56.18\%_{\pm 0.22\%}$	$\mathbf{56.61\%}_{\pm 0.23\%}$	$40.47\%_{\pm 1.16\%}$
B4TWR7	$50.13\%_{\pm 0.26\%}$	$\mathbf{50.30\%}_{\pm 0.14\%}$	$36.27\%_{\pm 0.80\%}$
Q91X51	$52.37\%_{\pm 0.16\%}$	$\mathbf{52.39\%}_{\pm 0.16\%}$	$40.47\%_{\pm 0.64\%}$
Q89BP2	$50.29\%_{\pm 0.13\%}$	$\mathbf{50.32\%}_{\pm 0.16\%}$	$39.41\%_{\pm 0.65\%}$
A6L9J9	$46.88\%_{\pm 0.14\%}$	$\mathbf{47.06\%}_{\pm 0.17\%}$	$36.42\%_{\pm 0.67\%}$
Q88X33	$\mathbf{42.55\%}_{\pm 0.27\%}$	$42.48\%_{\pm 0.28\%}$	$31.61\%_{\pm 1.65\%}$
B7KHU9	$\mathbf{40.61\%}_{\pm 0.33\%}$	$40.01\%_{\pm 0.57\%}$	$31.29\%_{\pm 1.47\%}$
Average	50.78%	**50.97%**	36.89%

Table 4: Results for the hypervolume indicator, in the format: $\text{median}_{\pm quartile\_deviation}$. Comparison among the three variants of MOSFLA. In bold we highlight the best results.

4.4. Comparison with the state of the art

As we mentioned at the beginning of this section, we have compared MOSFLA's results with the results of the multi-objective method from Terai et al. (2017), a previous proposal that is focused on the same multi-objective optimization problem. To get results for the nine instances (see Table 1) we have used their web-based application[3] with its default settings (those proposed by its authors). In the case of MOSFLA-2CH, the configuration was previously explained (see Section 4.1). The only change is the P_m that was slightly modified from 1.25 to 0.625 taking into account the new configuration experiments performed.

To evaluate the quality of the results and compare the methods, we apply three quality metrics widely used in multi-objective optimization: the hypervolume (Beume et al. (2009)), the set coverage (Zitzler et al. (2003)), and the maximum spread (Zitzler et al. (2000)). Since the Terai's method uses the three objectives previously defined, all the quality metrics are applied taking into account the three objectives, although our approach (MOSFLA-2CH) only optimizes two of them (mCAI and mHD, the third one is indirectly optimized, as it was shown in Section 4.3). In this way, the comparisons between both methods are fair, because both methods are compared in the same three-objective space. Furthermore, we have also performed different statistical analyses to evaluate and assure that the differences are statistically significant.

4.4.1. Hypervolume indicator

The hypervolume (HV) indicator has been calculated as it was previously explained. Table 5 shows the median HV results and their quartile deviations calculated for each instance. Also, the last row in the table presents the average HV. We can observe that MOSFLA-2CH obtains better hypervolume than the multi-objective method proposed by Terai et al., for all the instances. This means that the Pareto fronts generated by MOSFLA-2CH cover a greater portion of the objective space than the Pareto fronts obtained by Terai's method.

[3] `http://tandem.trahed.jp/tandem/`

Protein	MOSFLA-2CH	Terai et al. (2017)
Q5VZP5	**64.49%**$_{\pm 0.47\%}$	59.92%$_{\pm 0.18\%}$
A4Y1B6	**54.74%**$_{\pm 0.27\%}$	52.53%$_{\pm 0.06\%}$
B3LS90	**57.22%**$_{\pm 0.19\%}$	54.62%$_{\pm 0.16\%}$
B4TWR7	**50.54%**$_{\pm 0.14\%}$	48.91%$_{\pm 0.21\%}$
Q91X51	**52.73%**$_{\pm 0.17\%}$	50.47%$_{\pm 0.23\%}$
Q89BP2	**50.61%**$_{\pm 0.22\%}$	48.61%$_{\pm 0.22\%}$
A6L9J9	**47.06%**$_{\pm 0.22\%}$	45.56%$_{\pm 0.13\%}$
Q88X33	**42.32%**$_{\pm 0.25\%}$	41.07%$_{\pm 0.09\%}$
B7KHU9	**38.89%**$_{\pm 0.45\%}$	38.26%$_{\pm 0.12\%}$
Average	**50.96%**	48.88%

Table 5: Results for the hypervolume indicator, in the format: median$_{\pm quartile_deviation}$. Comparison between MOSFLA-2CH and Terai's method. In bold we highlight the best results.

This difference in hypervolume is also illustrated graphically when the Pareto fronts are displayed. As an example, the median Pareto fronts obtained for the protein $B3LS90$ are shown in Figure 3. We can observe that, in all the projections, the points (solutions) from MOSFLA-2CH are better and cover a greater region of the objective space.

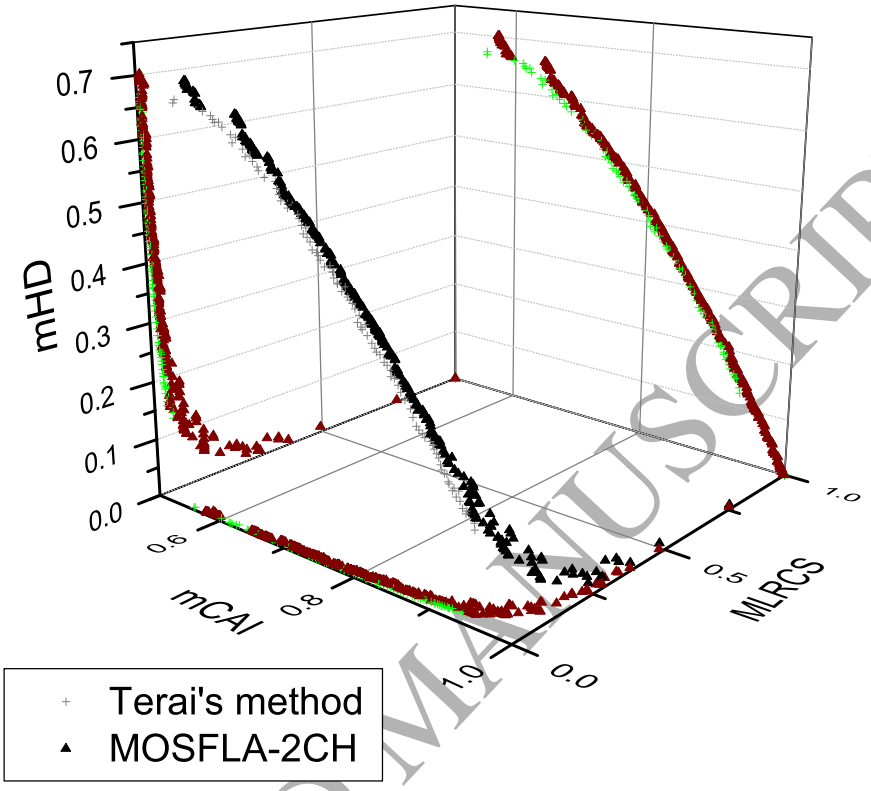


Figure 3: 3D scatter plot of the median Pareto fronts for the $B3LS90$ protein. Comparison between MOSFLA-2CH and Terai's method. The points in the different 2D projections appear in red (MOSFLA-2CH) or green (Terai's method), using the corresponding symbol.

4.4.2. Statistical significance in the hypervolume values

To ensure that the results relative to the HV indicator present statistically significant differences, we perform an statistical analysis with a significance level (p-value) of 0.05 or, in other words, a confidence level of 95%. In particular, a parametric test, ANalysis Of VAriance (ANOVA), will be applied, but this requires that the samples follow a normal distribution and they have homogeneous variances (homoscedasticity), so these features have to be checked before. In the case that these features are not fulfilled, we will apply a non-parametric test,

the Mann-Whitney U test. An exhaustive explanation about all these statistical tests can be found in Sheskin (2011).

The first test is the Kolmogorov-Smirnov test (KS test), checking that the samples follow a normal distribution. As the results in Table 6 show, three of the cases do not follow a normal distribution, therefore, being not feasible in theses cases to apply the ANOVA test. The second test is the Levene test. The samples that follow a normal distribution are checked about the homogeneity of their variances. As conclusion, in Table 6, we can see that there are four cases where the ANOVA test cannot be applied. As said, in these cases, we apply the Mann-Whitney U test.

Protein	**KS test**		**Levene test**	**Pass?**
	MOSFLA-2CH	**Terai et al. (2017)**		
Q5VZP5	0.000	0.200	–	No
A4Y1B6	0.006	0.200	–	No
B3LS90	0.200	0.148	0.247	Yes
B4TWR7	0.200	0.200	0.541	Yes
Q91X51	0.200	0.200	0.100	Yes
Q89BP2	0.200	0.200	0.606	Yes
A6L9J9	0.044	0.200	–	No
Q88X33	0.200	0.200	0.084	Yes
B7KHU9	0.078	0.200	0.046	No

Table 6: Normality analysis using the Kolmogorov-Smirnov test and homoscedasticity analysis using the Levene test. Both are about the hypervolume values.

The results from the ANOVA or the Mann-Whitney U tests are shown in Table 7. As we can see, all the differences in hypervolume between both methods are statistically significant, for all the instances. Therefore, the statistical analyses strengthen the conclusion that MOSFLA-2CH obtains better hypervolume results.

Protein	ANOVA test	Mann-Whitney U test	Statistical significance
Q5VZP5	–	0.000	Yes
A4Y1B6	–	0.000	Yes
B3LS90	0.000	–	Yes
B4TWR7	0.000	–	Yes
Q91X51	0.000	–	Yes
Q89BP2	0.000	–	Yes
A6L9J9	–	0.000	Yes
Q88X33	0.000	–	Yes
B7KHU9	–	0.001	Yes

Table 7: Results of the ANOVA or the Mann-Whitney U tests, depending on the previous tests, to find statistical significance in the hypervolume values.

4.4.3. Set coverage indicator

Set Coverage (SC) is the second indicator used to measure the quality of the results. In contrast to the previous indicator, it is a binary indicator. This measure is calculated by taking the Pareto front from each algorithm and counting how many solutions belonging to a Pareto front $\mathcal{B}$ are covered by at least one of the solutions from the other Pareto front $\mathcal{A}$. The calculation of this indicator is shown in Equation 8.

$$SC(\mathcal{A}, \mathcal{B}) = \frac{|\{b_j \in \mathcal{B}; \exists\ a_i \in \mathcal{A} : a_i \succeq b_j\}|}{|\mathcal{B}|}, \tag{8}$$

where $|\mathcal{B}|$ is the size (cardinality) of the set $\mathcal{B}$.

If for each solution in $\mathcal{B}$ there is at least a solution in $\mathcal{A}$ that covers it then SC($\mathcal{A}$,$\mathcal{B}$) is equal to 1. On the other hand, if none of the solutions in $\mathcal{B}$ is covered by any of the solutions from $\mathcal{A}$ then SC($\mathcal{A}$,$\mathcal{B}$) is 0. This indicator has to be calculated in both directions, SC($\mathcal{A}$,$\mathcal{B}$) and SC($\mathcal{B}$,$\mathcal{A}$), because a solution is considered covered when it is dominated or equal (weak dominance), therefore,

the indicator is not symmetric and it is possible that $SC(\mathcal{B}, \mathcal{A}) \neq 1 - SC(\mathcal{A}, \mathcal{B})$.

Table 8 shows the results obtained for the SC indicator. We can see that, in almost all the cases, the MOSFLA-2CH algorithm has a higher set coverage than the Terai's method. This implies that the Pareto fronts from MOSFLA-2CH cover a higher percentage of solutions from the Terai's method than conversely.

Protein	**SC(MOSFLA-2CH, Terai et al. (2017))**	**SC(Terai et al. (2017), MOSFLA-2CH)**
Q5VZP5	**19.00%**	8.43%
A4Y1B6	**27.00%**	4.36%
B3LS90	**14.00%**	1.02%
B4TWR7	**10.00%**	6.47%
Q91X51	**5.00%**	4.83%
Q89BP2	**11.00%**	5.83%
A6L9J9	5.00%	**6.83%**
Q88X33	4.00%	**7.83%**
B7KHU9	**32.00%**	8.83%
Average	**14.11%**	6.05%

Table 8: Results for the set coverage indicator. Comparison between MOSFLA-2CH and Terai's method. In bold we highlight the best results.

4.4.4. Maximum spread indicator

In this section, we apply a unitary indicator called Maximum Spread (MS), which measures the distribution extension of the non-dominated solutions set in the objective space (Pareto front). It is based on the minimum and maximum scores from each objective function as shown in Equation 9.

$$MS = \sqrt{\sum_{m=1}^{M} (max_{i=1}^{NDS} f_m^i - min_{i=1}^{NDS} f_m^i)^2}, \tag{9}$$

where M defines the number of objective functions and NDS indicates the size of the non-dominated solutions set.

The measure has been calculated for both methods and for the 9 problem instances. Table 9 collects the median value and quartile deviation of the maximum spread. Also, the last row of the table contains the average value taking into account all the instances. We can observe that MOSFLA-2CH obtains better results in almost all the instances, with the only exception of the protein $B4TWR7$ (whose results are very similar). In fact, MOSFLA-2CH also obtains better average maximum spread. In conclusion, we can say that the Pareto fronts generated by MOSFLA-2CH have better spread, with their edges being farther apart one from the other.

Protein	**MOSFLA-2CH**	**Terai et al. (2017)**
Q5VZP5	$\mathbf{1.691}_{\pm 0.004}$	$1.349_{\pm 0.005}$
A4Y1B6	$\mathbf{1.292}_{\pm 0.003}$	$1.284_{\pm 0.001}$
B3LS90	$\mathbf{1.285}_{\pm 0.002}$	$1.278_{\pm 0.002}$
B4TWR7	$1.263_{\pm 0.003}$	$\mathbf{1.264}_{\pm 0.001}$
Q91X51	$\mathbf{1.314}_{\pm 0.003}$	$1.309_{\pm 0.001}$
Q89BP2	$\mathbf{1.290}_{\pm 0.002}$	$1.286_{\pm 0.003}$
A6L9J9	$\mathbf{1.249}_{\pm 0.003}$	$1.242_{\pm 0.003}$
Q88X33	$\mathbf{1.162}_{\pm 0.004}$	$1.158_{\pm 0.004}$
B7KHU9	$\mathbf{1.199}_{\pm 0.006}$	$1.184_{\pm 0.005}$
Average	**1.305**	1.262

Table 9: Results for the maximum spread indicator, in the format: $\text{median}_{\pm quartile\_deviation}$. Comparison between MOSFLA-2CH and Terai's method. In bold we highlight the best results.

4.4.5. Statistical significance in the maximum spread values

To make sure that the maximum spread differences are statistically significant, we will apply the ANOVA test. The whole statistical analysis has been performed with a confidence level of 95% (significance level or p-value of 0.05).

Before applying the ANOVA test, we have to verify that the samples follow a normal distribution and their variances are homogeneous (homoscedasticity). In case these properties are not fulfilled, we will apply the Mann-Whitney U test.

Table 10 shows the results of the Kolmogorov-Smirnov test (KS test, checking that the samples follow a normal distribution) and the Levene test (checking the homoscedasticity). As we can see, in two cases the samples do not follow a normal distribution, and furthermore, in another case the samples do not have the homoscedasticity property. Therefore, these three cases are analyzed by using the Mann-Whitney U test, and the rest by using the ANOVA test.

Protein	**KS test**		**Levene test**	**Pass?**
	MOSFLA-2CH	**Terai et al. (2017)**		
Q5VZP5	0.000	0.200	–	No
A4Y1B6	0.200	0.200	0.234	Yes
B3LS90	0.200	0.200	0.843	Yes
B4TWR7	0.200	0.200	0.866	Yes
Q91X51	0.200	0.032	–	No
Q89BP2	0.098	0.200	0.025	No
A6L9J9	0.200	0.200	0.747	Yes
Q88X33	0.200	0.200	0.098	Yes
B7KHU9	0.200	0.200	0.730	Yes

Table 10: Normality analysis using the Kolmogorov-Smirnov test and homoscedasticity analysis using the Levene test. Both are about the maximum spread values.

Finally, Table 11 shows the cases that have a statistically significant difference between both methods. As a result, we can see that the differences are statistically significant in almost all the cases, with the exception of proteins $B4TWR7$ and $Q88X33$. Observe that the protein $B4TWR7$ was the only case in which the Terai's method obtained better maximum spread (see Table 9).

Therefore, the statistical analyses strengthen the conclusion that MOSFLA-2CH obtains better maximum spread results.

Protein	ANOVA test	Mann-Whitney U test	Statistical significance
Q5VZP5	–	0.000	Yes
A4Y1B6	0.000	–	Yes
B3LS90	0.002	–	Yes
B4TWR7	0.430	–	No
Q91X51	–	0.019	Yes
Q89BP2	–	0.047	Yes
A6L9J9	0.005	–	Yes
Q88X33	0.744	–	No
B7KHU9	0.004	–	Yes

Table 11: Results of the ANOVA or the Mann-Whitney U tests, depending on the previous tests, to find statistical significance in the maximum spread values.

5. Conclusions and Future Work

In this work, we have designed and implemented a method for designing CDSs encoding the same protein, an important problem in synthetic biology. This problem is a multi-objective optimization problem that, taking into account the previous literature, involves three objective functions: mCAI (Codon Adaptation Index), mHD (Hamming Distance between CDSs), and MLRCS (Length of Repeated or Common Substrings). One of the contributions of this work is that this multi-objective problem can be re-defined, using only two objectives: mCAI and mHD. In fact, our approach only optimizes these two objectives. This is possible because we have shown that the third objective (MLRCS) is correlated with mHD, and optimizing mHD also implies the optimization of MLRCS.

In our proposal, we have adapted SFLA, a memetic meta-heuristic algorithm

based on swarm intelligence, to the particular needs of this multi-objective optimization problem and we have designed MOSFLA-2CH. In order to evaluate the quality of our approach, we have compared it with the Terai's method. A multi-objective method proposed recently (2017), which has the same purpose, that is, addressing the same problem. The comparisons have been performed by using 9 real protein instances. These instances have been selected in order to cover a wide range of situations (with different amino acid lengths and numbers of CDSs), therefore, being a representative sample. Three typical quality metrics in multi-objective optimization have been used in the comparisons. These metrics analyze different important aspects in multi-objective optimization, such as convergence, uniformity, and spread. The comparisons show that MOSFLA-2CH attains better results than the Terai's method in almost all the instances. Furthermore, after a comprehensive statistical analysis this conclusion has been strengthened, finding statistically significant differences between both methods.

MOSFLA-2CH's better results denote its higher ability for processing complex search spaces. More specifically: 1) it conducts multiple, simultaneous searches over different memeplexes (sets of solutions); 2) it generates new solutions by taking as reference the information provided by the best local solution within the processed memeplex and the best global solution in the population, also addressing stagnation situations by re-initializing solutions; and 3) it uses shuffling techniques to achieve a global exchange of knowledge among memeplexes, allowing a balanced distribution of promising solutions for improved optimization purposes.

Regarding previous expert and intelligent systems based on SFLA, Niknam et al. (2011) proposed a Chaotic Modified Shuffled Frog Leaping Algorithm (CMSFLA) for solving the economic dispatch problem. CMSFLA changes the standard local search in each memeplex by a chaotic local search. Luo & Chen (2014) presented an improved Shuffled Frog Leaping Algorithm (SFLA) to solve the multi-depot vehicle routing problem. In particular, they improved SFLA by using a novel real number encoding method specially adapted to the multi-

depot vehicle routing problem. Luo et al. (2014) designed a Modified Shuffled Frog Leaping Algorithm (MSFLA) for solving the dynamic allocation problem of virtual machines in cloud data centers. MSFLA includes a new parameter w, the leaping vision weight. This parameter is dynamically adjusted to go from a global exploration to a local exploitation as the iteration progresses. Zhu & Zhang (2014) implemented an improved Shuffled Frog Leaping Algorithm to solve the component pick-and-place sequencing problem. More specifically, SFLA was improved with the strategy of letting all frogs taking part in memetic evolution. Finally, Lei & Guo (2015) proposed a modified Shuffled Frog Leaping Algorithm for solving the two-agent hybrid flow shop scheduling problem. Their modified SFLA includes a tournament selection based method to divide population, that is, not all solutions of population are allocated into memeplexes. Furthermore, in the search process within each memeplex, all solutions in the memeplex can be used with the same probability, and not only the worst local solution in every moment. Our proposal, MOSFLA, is clearly different to all these previous ones, including new important theoretical contributions. Firstly, as our problem is a multi-objective optimization problem, we have adapted SFLA with a multi-objective algorithmic design. More specifically, MOSFLA includes a multi-objective, quality-aware population partitioning to conduct multiple, simultaneous searches over different sets of solutions at each evolutionary step. Moreover, this multi-objective algorithmic design implies the comparison of solutions based on the dominance concept, the use of non-dominated sortings, and the application of the crowding multi-objective metric (a diversity metric). And secondly, MOSFLA includes both standard operators and expert, problem-oriented mutations (a total of four operators are integrated into its algorithmic design) to effectively deal with the different objectives defined in the formulation of the problem.

As future work, we have planned to study other multi-objective algorithms to compare them with MOSFLA-2CH, allowing us to further assess the good quality of the MOSFLA-2CH results or even improve these results. As a second research line, due to the good results obtained by MOSFLA, we intend to apply

this multi-objective algorithm to other bioinformatics multi-objective problems, assessing if MOSFLA also obtains good results in these other problems.

CRediT author statement

Belen Gonzalez-Sanchez: Software, Validation, Formal Analysis, Investigation, Data Curation, Writing - Original Draft, Visualization.
Miguel A. Vega-Rodrguez: Conceptualization, Methodology, Resources, Writing - Review & Editing, Supervision, Project Administration, Funding Acquisition.
Sergio Santander-Jimnez: Conceptualization, Writing - Review & Editing, Funding Acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partially funded by the AEI (State Research Agency, Spain) and the ERDF (European Regional Development Fund, EU), under the contract TIN2016-76259-P (PROTEIN project), as well as funds through FCT (Fundação para a Ciência e a Tecnologia, Portugal) with reference UID/CEC/50021/2019. Thanks also to the Junta de Extremadura and the ERDF for the grants GR18090 and IB16002 provided to the research group TIC015. Sergio Santander-Jiménez is supported by the Post-Doctoral Fellowship SFRH/BPD/119220/2016 from FCT.

References

Athey, J., Alexaki, A., Osipova, E., Rostovtsev, A., Santana-Quintero, L. V., Katneni, U., Simonyan, V., & Kimchi-Sarfaty, C. (2017). A new and updated resource for codon usage tables. *BMC Bioinformatics*, *18*, 391. doi:10.1186/s12859-017-1793-7.

Aw, R., & Polizzi, K. M. (2013). Can too many copies spoil the broth? *Microbial Cell Factories*, *12*, 128. doi:10.1186/1475-2859-12-128.

Bartle, R. G. (1995). *The Elements of Integration and Lebesgue Measure*. John Wiley & Sons, Inc. doi:10.1002/9781118164471.

Beume, N., Fonseca, C. M., Lopez-Ibanez, M., Paquete, L., & Vahrenhold, J. (2009). On the complexity of computing the hypervolume indicator. *IEEE Transactions on Evolutionary Computation*, *13*, 1075–1082. doi:10.1109/tevc.2009.2015575.

Chen, S., Li, K., Cao, W., Wang, J., Zhao, T., Huan, Q., Yang, Y.-F., Wu, S., & Qian, W. (2017). Codon-resolution analysis reveals a direct and context-dependent impact of individual synonymous mutations on mRNA level. *Molecular Biology and Evolution*, *34*, 2944–2958. doi:10.1093/molbev/msx229.

Chin, J. X., Chung, B. K.-S., & Lee, D.-Y. (2014). Codon Optimization OnLine (COOL): a web-based multi-objective optimization platform for synthetic gene design. *Bioinformatics*, *30*, 2210–2212. doi:10.1093/bioinformatics/btu192.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*, 182–197. doi:10.1109/4235.996017.

Elbeltagi, E., Hegazy, T., & Grierson, D. (2007). A modified shuffled frog-leaping optimization algorithm: applications to project manage-

ment. *Structure and Infrastructure Engineering*, *3*, 53–60. doi:10.1080/15732470500254535.

Gu, P., Yang, F., Su, T., Wang, Q., Liang, Q., & Qi, Q. (2015). A rapid and reliable strategy for chromosomal integration of gene(s) with multiple copies. *Scientific Reports*, *5*, 9684. doi:10.1038/srep09684.

Guimaraes, J. C., Rocha, M., Arkin, A. P., & Cambray, G. (2014). D-Tailor: automated analysis and design of DNA sequences. *Bioinformatics*, *30*, 1087–1094. doi:10.1093/bioinformatics/btt742.

Hohenblum, H., Gasser, B., Maurer, M., Borth, N., & Mattanovich, D. (2004). Effects of gene dosage, promoters, and substrates on unfolded protein stress of recombinant Pichia pastoris. *Biotechnology and Bioengineering*, *85*, 367–375. doi:10.1002/bit.10904.

Hu, B., Dai, Y., Su, Y., Moore, P., Zhang, X., Mao, C., Chen, J., & Xu, L. (2018). Feature selection for optimized high-dimensional biomedical data using the improved shuffled frog leaping algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, (pp. 1–10). doi:10.1109/TCBB.2016.2602263.

Khasanov, F. K., Zvingila, D. J., Zainullin, A. A., Prozorov, A. A., & Bashkirov, V. I. (1992). Homologous recombination between plasmid and chromosomal DNA in Bacillus subtilis requires approximately 70 bp of homology. *Molecular and General Genetics*, *234*, 494–497. doi:10.1007/BF00538711.

Lei, D., & Guo, X. (2015). A shuffled frog-leaping algorithm for hybrid flow shop scheduling with two agents. *Expert Systems with Applications*, *42*, 9333–9339. doi:10.1016/j.eswa.2015.08.025.

Lin, J., Zhong, Y., & Zhang, J. (2012). A modified discrete shuffled flog leaping algorithm for RNA secondary structure prediction. In D. Zeng (Ed.), *Advances in Control and Communication. LNEE, vol. 137* (pp. 591–599). Berlin,

Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-642-26007-0_73.

Luo, J., & Chen, M.-R. (2014). Improved shuffled frog leaping algorithm and its multi-phase model for multi-depot vehicle routing problem. *Expert Systems with Applications*, *41*, 2535–2545. doi:10.1016/j.eswa.2013.10.001.

Luo, J., Li, X., & Chen, M.-R. (2014). Hybrid shuffled frog leaping algorithm for energy-efficient dynamic consolidation of virtual machines in cloud data centers. *Expert Systems with Applications*, *41*, 5804–5816. doi:10.1016/j.eswa.2014.03.039.

Manivasakam, P., Weber, S. C., McElver, J., & Schiestl, R. H. (1995). Micro-homology mediated PCR targeting in Saccharomyces cerevisiae. *Nucleic Acids Research*, *23*, 2799–2800. doi:10.1093/nar/23.14.2799.

Niknam, T., Firouzi, B. B., & Mojarrad, H. D. (2011). A new evolutionary algorithm for non-linear economic dispatch. *Expert Systems with Applications*, *38*, 13301–13309. doi:10.1016/j.eswa.2011.04.151.

Puigbò, P., Guzmán, E., Romeu, A., & Garcia-Vallvé, S. (2007). OPTIMIZER: a web server for optimizing the codon usage of DNA sequences. *Nucleic Acids Research*, *35*, W126–W131. doi:10.1093/nar/gkm219.

Sahoo, S., Das, S. S., & Rakshit, R. (2019). Codon usage pattern and predicted gene expression in Arabidopsis thaliana. *Gene: X*, *2*, 100012. doi:10.1016/j.gene.2019.100012.

Sarkheyli, A., Zain, A. M., & Sharif, S. (2015). The role of basic, modified and hybrid shuffled frog leaping algorithm on optimization problems: a review. *Soft Computing*, *19*, 2011–2038. doi:10.1007/s00500-014-1388-4.

Scorer, C. A., Clare, J. J., McCombie, W. R., Romanos, M. A., & Sreekrishna, K. (1994). Rapid selection using G418 of high copy number transformants of Pichia pastoris for high-level foreign gene expression. *Bio/Technology*, *12*, 181–184. doi:10.1038/nbt0294-181.

Sharp, P. M., & Li, W.-H. (1987). The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Research*, *15*, 1281–1295. doi:10.1093/nar/15.3.1281.

Shen, P., & Huang, H. V. (1986). Homologous recombination in Escherichia coli: dependence on substrate length and homology. *Genetics*, *112*, 441–457.

Sheskin, D. J. (2011). *Handbook of Parametric and Nonparametric Statistical Procedures*. (5th ed.). NY, USA: Chapman & Hall/CRC.

Song, H., Gao, H., Liu, J., Tian, P., & Nan, Z. (2017). Comprehensive analysis of correlations among codon usage bias, gene expression, and substitution rate in Arachis duranensis and Arachis ipaënsis orthologs. *Scientific Reports*, *7*, 14853. doi:10.1038/s41598-017-13981-1.

Terai, G., Kamegai, S., Taneda, A., & Asai, K. (2017). Evolutionary design of multiple genes encoding the same protein. *Bioinformatics*, *33*, 1613–1620. doi:10.1093/bioinformatics/btx030.

Tran, T.-A., Vo, N. T., Nguyen, H. D., & Pham, B. T. (2015). A novel method to predict highly expressed genes based on radius clustering and relative synonymous codon usage. *Journal of Computational Biology*, *22*, 1086–1096. doi:10.1089/cmb.2015.0121.

Tyo, K. E. J., Ajikumar, P. K., & Stephanopoulos, G. (2009). Stabilized gene duplication enables long-term selection-free heterologous pathway expression. *Nature Biotechnology*, *27*, 760–765. doi:10.1038/nbt.1555.

Vasanthi, S., & Dass, J. F. P. (2018). Comparative genome-wide analysis of codon usage of different bacterial species infecting Oryza sativa. *Journal of Cellular Biochemistry*, *119*, 9346–9356. doi:10.1002/jcb.27214.

Vassileva, A., Chugh, D. A., Swaminathan, S., & Khanna, N. (2001). Expression of hepatitis B surface antigen in the methylotrophic yeast Pichia pastoris using the GAP promoter. *Journal of Biotechnology*, *88*, 21–35. doi:10.1016/S0168-1656(01)00254-1.

Wang, W., Cheng, X., Buske, P. J., Suzich, J. A., & Jin, H. (2019). Attenuate Newcastle disease virus by codon modification of the glycoproteins and phosphoprotein genes. *Virology*, *528*, 144–151. doi:10.1016/j.virol.2018.12.017.

Webster, G. R., Teh, A. Y.-H., & Ma, J. K.-C. (2017). Synthetic gene design - The rationale for codon optimization and implications for molecular pharming in plants. *Biotechnology and Bioengineering*, *114*, 492–502. doi:10.1002/bit.26183.

Yu, C.-H., Dang, Y., Zhou, Z., Wu, C., Zhao, F., Sachs, M. S., & Liu, Y. (2015). Codon usage influences the local rate of translation elongation to regulate co-translational protein folding. *Molecular Cell*, *59*, 744–754. doi:10.1016/j.molcel.2015.07.018.

Zhang, R., Zhang, L., Wang, W., Zhang, Z., Du, H., Qu, Z., Li, X.-Q., & Xiang, H. (2018). Differences in codon usage bias between photosynthesis-related genes and genetic system-related genes of chloroplast genomes in cultivated and wild Solanum species. *International Journal of Molecular Sciences*, *19*, 3142. doi:10.3390/ijms19103142.

Zhu, G.-Y., & Zhang, W.-B. (2014). An improved shuffled frog-leaping algorithm to optimize component pick-and-place sequencing optimization problem. *Expert Systems with Applications*, *41*, 6818–6829. doi:10.1016/j.eswa.2014.04.038.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: empirical results. *Evolutionary Computation*, *8*, 173–195. doi:10.1162/106365600568202.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & da Fonseca, V. G. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, *7*, 117–132. doi:10.1109/tevc.2003.810758.