Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 337–346
DOI: 10.15439/2022F21
ISSN 2300-5963 ACSIS, Vol. 30

A Multi-objective Cluster-based Biased Random-Key Genetic Algorithm with Online Parameter Control Applied to Protein Structure Prediction

Felipe Marchi
Santa Catarina State University
Applied Computing Graduate Program
Joinville, SC - Brazil
Email: felipe.r.marchi@gmail.com

Rafael Stubs Parpinelli
Santa Catarina State University
Applied Computing Graduate Program
Joinville, SC - Brazil
Email: rafael.parpinelli@udesc.br

***Abstract*—The protein structure prediction problem is one of the most important bioinformatics problems. Computational methods can be used to approach this problem and *de novo* methods are able to generate protein structures without the need of having known similar structures to the predicted protein. These methods transform the prediction problem into an optimization problem, using optimization models that combine different energy functions and high-level information. These models usually have only a single optimization objective. However, it is known that this single objective optimization approach may harm the optimization search due to the existence of conflicts between the different terms that compose the optimization objective. The proposed model has three objectives: energy function, secondary structure, and contact maps. A multi-objective Biased Random-Key Genetic Algorithm (BRKGA) with online parameter control, named MOBO, is proposed as the optimizer. The final predictor comprises two phases of the MOBO algorithm and selects a final structure using the MUFOLD-CL clustering method. Results obtained demonstrated that the proposed predictor generated highly competitive results with the literature.**

***Index Terms*—Bioinformatics, Multi-objective Optimization, Evolutionary Computation, Clustering, Online Parameter Tuning, Protein Structure Prediction**

I. Introduction

PROTEINS are base molecules present in living organisms [1]. They are responsible for many biological functions, and understanding their mechanisms is important to better understand how organisms work. One application of this knowledge about proteins is in the biomedicine and pharmaceutics areas [2], where novel proteins could be developed to combat particular types of diseases.

The Protein Structure Prediction (PSP) problem has the objective of determining the spatial structure of proteins, using the amino acid sequence of a protein as its main input. Although these structures can be determined using classical laboratory methods, such as Nuclear Magnetic Resonance (NMR) or X-ray crystallography [3], they have a considerable cost and are time-consuming. An alternative method is the use of computational resources to simulate these structures.

Recently, the AlphaFold, the deep learning algorithm developed by DeepMind, achieved highly promising results in the CASP13 and CASP14 competition [4]. The AlphaFold combines features derived from homologous templates and from multiple sequence alignment to generate the predicted structure. Nevertheless, AlphaFold has some drawbacks, such as the bias to the PDB database, and the heavy dependency on high-computational efforts for training their model [5]. Also, given the number of possible natural and artificial protein structures, it is currently unfeasible to rely on template-based methods to predict any unknown structure with consistent quality. As such, the PSP is still considered to be an open problem. So far, there is no viable general solution to this challenging problem. In this way, metaheuristics can be a faster option to the problem even though achieved results are not the same as AlphaFold.

In the literature, there are many computational methods proposed as possible approaches to the PSP problem [6], [7]. Among these methods, some works further explore the use of specific types of protein structure information [8], [9].

For a more general solution to this problem, one possible way is to use methods that do not rely on databases. One class of methods that do not employ known structure information are the *ab initio* methods. These methods work by generating structures using some evaluation function, which guides the optimization towards the optimal structure. Different from other types of methods, these methods only need as input the amino acid sequence to work [3].

While it is possible to work with pure *ab initio* methods, which utilize only information provided by the amino acid sequence, their results may be unfeasible given inaccuracies of currently employed energy functions. One way to deal with this issue is to utilize high-level information about the protein structure, such as secondary structure and contact map information. *Ab initio* methods that employ this type of information are called *de novo* methods. This work employs *de novo* methods instead of pure *ab initio*.

All this information related to the PSP problem can be described in an objective function. However, combining multiple information in a single objective function is not always optimal. It is known that some of the terms that compose an energy function, such as the bonded and non-bonded energies, are in conflict [10]. These conflicts indicate that optimizing one particular function term may not optimize the others. Hence, it is interesting to separate conflicting terms in different objectives. The separation of conflicting terms creates a multi-objective model, whose final solution is a set of non-dominating solutions [11]. This set, also called the *Pareto set*, represents possible solutions for the mathematical model.

As the optimization result of a multi-objective problem is a set of non-dominating solutions [11], it may be necessary to employ a decision-making method to select a single final solution from this set. As each non-dominated solution represents some trade-off among all objectives, it is not always trivial to choose a singular solution. To assist this selection, decision-making methods can be employed to filter the non-dominated set and identify the most promising solutions based on some desired properties.

This work is an extension of a previous study [12]. The previous work proposed the use of the Biased Random-Key Genetic Algorithm (BRKGA) method as an optimizer for a *de novo* multi-objective optimization model of the PSP problem. The present work brings some contributions compared to the previous publication:

- Online parameter control strategies were incorporated, creating the new version called MOBO. The main objective of this change is to reduce the number of parameters that the user must define to use the optimizer effectively. This change makes it possible to dedicate more time to problem modeling than parameter tuning.
- The MUFOLD-CL was employed as decision-maker in the Pareto set. The generated clusters were analyzed to determine the distance between the best solution found by the optimizer and the solution selected by the decision-maker.

II. Background

A. Proteins and Protein Structure Prediction

Proteins are biomolecules composed of amino acids linked by peptide bonds. The amino acids are molecules composed of a central carbon, called α-carbon, that is connected to an amino group (NH), a carboxyl group (CO), a hydrogen molecule (H), and a side chain [1]. The side chain is a variable group unique to each type of amino acid. The molecular structure of each amino acid is the same, changing only the side chain, which gives its identity.

The structure of a generic amino acid can be seen in Figure 1. These amino acids connect between themselves through the peptide bonds, created by the combination of the carboxyl and amino group [1]. A point of interest in this figure is the three dihedral angles (Φ, Ψ, and Ω), which are important for computational representation of the protein structures.

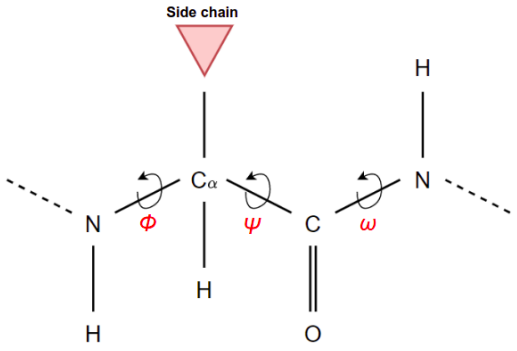


Fig. 1: Chemical structure of an amino acid

1) Computational representation: There are several ways to represent a protein structure computationally. The most direct would be to maintain complete information on all the atoms and interactions of the structure. However, this type of representation can be computationally expensive given the large number of atoms and interactions of a single protein structure [3].

One interesting computational representation of protein structures uses only the backbone and side chain torsion angles of each amino acid. Instead of maintaining the entire atomic structure of each amino acid, algorithms may use only the ϕ, ψ, and ω backbone angles and the χ side chain angles. These angles can be seen in Figure 1. By using these angles, the final three-dimensional structure can be uniquely determined.

2) Structure prediction: Computational methods can be used to generate protein structures. One class of such methods are the *ab initio* methods, which do not use information from known protein structures [3]. These methods use physical energy functions to guide the generation of protein structures, using only the amino acid as necessary input. As such, *ab initio* methods effectively transform the PSP problem into an optimization problem.

Besides the computational representation of a protein structure, defining a function to evaluate these structures is also necessary. As *ab initio* methods generally perform searches in the space of structures, it is necessary to select a suitable evaluation function. Regarding accuracy, the most optimal choice would be an evaluation function representing the natural energy function. However, such realistic energy functions are computationally expensive [2].

Other types of information can be incorporated into the evaluation function to enhance the search process. Although pure *ab initio* methods do not use information from known structures, it may be beneficial to use this type of information, if available. There are different types of high-level information that can be added to the evaluation function, such as *secondary structure prediction* and *contact maps*. When an *ab initio* method incorporates information other than the energy function, it is called an *de novo* method [3].

B. Biased Random-Key Genetic Algorithm

The Biased Random-Key Genetic Algorithm (BRKGA) [13] is an optimization method of the evolutionary algorithms class. It is a variation of the genetic algorithm in which only the selection and crossover routines are used to evolve solutions [13]. Initially, all individuals are initialized uniformly at random. Each generation, pairs of individuals are selected and combined through a biased uniform crossover operator, where one parent is an elite solution, and the other is a non-elite. A new population is formed by a fraction of the most-fit solutions of the current generation, a fraction of randomly and uniformly generated solutions, and the remaining individuals are the offspring generated through the crossover operation. There is no explicit mutation operator in this algorithm, but an implicit mutation can be simulated through the crossover of a random individual with a non-random individual.

The main aspect of the BRKGA is the clear separation of the problem-dependent and independent parts. Other notable aspects are elitism as a core mechanism for the evolution and the generation of random solutions instead of applying mutation to offspring solutions.

The problem-independent codification is composed of real numbers in the $[0, 1]$ interval, decoded into a problem-specific solution using a decoding function, and evaluated by a fitness function. This way, the algorithm's main structure is standardized for all problems, with only the decoding and fitness functions needing to be developed. This standardization simplifies the algorithm structure by removing the necessity of developing complex solution encodings and various genetic operators, such as crossover or mutation operators.

Similar to other evolutionary algorithms, the BRKGA has a set of parameters that need to be defined. These parameters are the number of iterations (N_{it}), population size (p), elite fraction (p_e), random individuals fraction (p_m), and crossover probability (c_{pr}). Due to the standardized structure of the algorithm, the crossover operator is fixed.

III. Related Work

One of the initial works in the area of multi-objective PSP was [10]. In this pioneering study, the authors proposed a multi-objective model for the PSP problem, decomposing the CHARMM energy function into two objectives: bonded and non-bonded energies. They demonstrated that these energies were in conflict, and combining them in a single objective would harm the optimization. To optimize the model, they used the IPAES algorithm and also employed a decision-making step to select a final structure by using the method of *knees*.

Other works followed this approach of breaking the energy function into bonded and non-bonded energies [14], [15], [16], [17], [18]. Most works approaching the multi-objective PSP employed some evolutionary algorithm [14], [15], [16], [19], [20], [18], [21]. Regarding the protein structure representation, most works employed the full atomic structure [14], [15], [16], [22], [21], although some preferred the centroid representation [19], [20], [18].

Considering the extra information that can be added to complement the energy function, there are several possibilities. Some works employ structural information that energy functions may not capture accurately, such as solvent terms [15], [16], [22], compactness [19], and hydrogen bonding [19], [20]. Also, high-level information that can be predicted was explored, such as protein fragments [19], [20], secondary structures [14], [15], [16], [17], [19], [22], [18], [21], and contact maps [19], [20], [22].

As the complete solution of multi-objective problems is a set of non-dominated solutions, selecting a single final solution from this set may be necessary. Several works employ some decision-making criteria to select the final optimized solution. Popular methods include the *knees* method [10], [23], [14] and clustering-based methods [15], [17], [17], [16], [21]. One promising clustering method for protein structures is the MUFOLD-CL method, which was employed in recent works for the multi-objective PSP problem [17], [21].

IV. Proposed Model

A multi-objective Biased Random-Key Genetic Algorithm (BRKGA) with online parameter control is proposed, named MOBO. In the present work, proteins are modeled with the Rosetta framework [1] using the full-atom with centroid backbone representation [24], where the side chain is simplified as a centroid. The ϕ, ψ, and ω angles model each amino acid. These angles are in the $[-180, 180]$ domain, except for the ω angle, whose optimal value can be either 180º or 0º [10]. In this work, the ω angle is restricted to 180º, as both values are equally optimal. Figure 2 shows how an amino acid sequence can be coded as a list of angles.

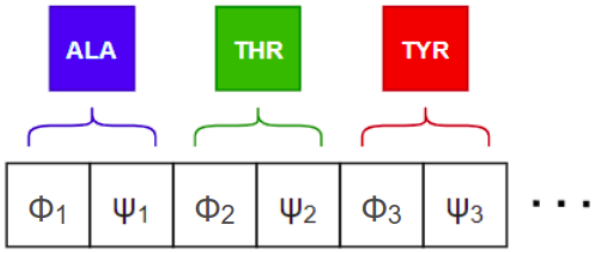


Fig. 2: Protein structure representation as a list of torsion angles.

The optimization model for the PSP problem is composed of 3 objectives, shown in Expressions (1)-(3):

$$\min \quad score4(\vec{x}) \tag{1}$$
$$\max \quad SS(\vec{x}) \tag{2}$$
$$\max \quad CM(\vec{x}) \tag{3}$$

The $\vec{x}$ vector represents a protein structure coded as a list of angles with size $2L$, where L is the length of the amino

[1] https://www.rosettacommons.org/software

acid sequence of the protein. The angle list is mapped into a protein structure before being evaluated by each objective function. Expression (1) is the *score4* energy function from the Rosetta framework [24] and is used to evaluate the protein structure.

Expression (2) refers to secondary structure information predicted by the PSIPRED [25] server [2]. This prediction was performed excluding homologous proteins. Equation (4) details the evaluation, where for each amino acid x_i, the pSS function will return the predicted probability for the current secondary structure manifested, determined with the DSSP program [26], which assigns secondary structure to proteins structures. Hence, protein structures with the most probable secondary structures are benefited.

$$SS(\vec{x}) = \sum_{i}^{L} pSS(x_i) \tag{4}$$

Expression (3) refers to contact map information predicted by the RaptorX [27] server [3]. This prediction was performed by removing homologous proteins. Two atoms are considered in contact if their distance is less than the cutoff distance. Equation (5) describes how each contact is evaluated for a given protein structure.

For each contact c_i in the L-best list, the pair of amino acids c_i^1 and c_i^2 from this contact is verified. If their distance, $d(c_i^1, c_i^2)$, is less than the cutoff value (8Å), the pair is said to be in contact, and a term equal to the predicted probability of this contact is summed. Otherwise, this probability term is decreased exponentially due to the cutoff value's distance, allowing slight deviations from the cutoff to be considered. Protein structures that exhibit the most probable contacts are rewarded using this objective function.

$$CM(\vec{x}) = \sum_{i}^{L} \begin{cases} p_i & \text{if} \quad d(c_i^1, c_i^2) \leq 8 \\ \frac{p_i}{e^{d(c_i^1, c_i^2)-8}} & \text{otherwise} \end{cases} \tag{5}$$

As global optimizer, the multi-objective BRKGA is employed[12]. The proposed algorithm uses the base structure of the original BRKGA [13] and can be applied to any multi-objective problem that can be approached with evolutionary algorithms. The main modifications were to allow the algorithm to optimize multi-objective problems. Modifications were made to the problem-independent parts to achieve multi-objective optimization. The elitist selection operator from NSGA-II [28] was used to generate the elite part of the population.

An archive (set of solutions) is maintained and updated at each generation. This archive has the same size as the population, and it is updated using the non-dominated sorting with crowding. At the end of the algorithm, the archive is returned and contains the best Pareto set found. However, it may also contain dominated solutions as the best Pareto set may not occupy the entire archive.

[2]http://bioinf.cs.ucl.ac.uk/psipred/

[3]http://raptorx.uchicago.edu/

A. Parameter control

There are several types of parameter control in evolutionary algorithms [29]. This work employs an adaptive online parameter control by simple rules to control and guide the optimizer. The information utilized to guide the search is the population diversity defined in Equation (6):

$$Div(P) = \frac{\sum_{x,y \in P} d_{norm}(x, y)}{|P| * (|P| - 1)} \tag{6}$$

where x and y are individuals from population P, and d_{norm} is the normalized Euclidean distance metric, defined by:

$$d_{norm}(\vec{x}, \vec{y}) = \frac{d(\vec{x}, \vec{y})}{N} \tag{7}$$

where $d(\vec{x}, \vec{y})$ is the Euclidean distance and N is a normalization factor representing the solution space diagonal. This distance is a genotypic diversity metric, that is, it measures the diversity in the problem-independent part of the algorithm. This work will not incorporate fitness information in the parameter control, as the definition of fitness is different for a multi-objective optimizer, and concepts such as best and mean fitness are not trivially defined.

The algorithm is modified to control and increase diversity in the population to incorporate diversity information. In the original BRKGA method, the crossover probability p_{cr} is defined as a parameter of the algorithm and should have a value in the range $(0.5, 1)$. In this work, the crossover parameter is modified to be a uniformly random value in the range $(0.5, 1)$, removing the complexity of defining an optimal value for this parameter and minimizing the impact in the diversity. This value is generated for each crossover operation.

Another modification is the introduction of two diversity control values: the diversity fraction (p_d) and the exploration diversity threshold (δ). These values are used to control the generation of the elite part of the population. The diversity fraction indicates the fraction of the elite part that should be diversified. The exploration diversity threshold is used to indicate the minimum distance between two diversified solutions during the exploration part of the search.

The set P_d of diversified elite solutions is defined as:

$$P_d = \{ \forall \vec{x} \in P_d \mid \Delta(\vec{x}) \geq \delta \} \tag{8}$$

where $\Delta(\vec{x})$ is defined as:

$$\Delta(\vec{x}) = \min_{\substack{\vec{y} \in P_d \\ \vec{x} \neq \vec{y}}} d_{norm}(\vec{x}, \vec{y}) \tag{9}$$

where d_{norm} is the normalized Euclidean distance expressed by Equation (7). This definition of diversified individuals spreads the solutions through the space. Incorporating this diversification in the elite part allows the algorithm to explore the best solutions in different parts of the search space.

Given the population of size p, $E = p \times p_e$ is the size of the elite part of the population and $E_d = E \times p_d$ is the size of the diversified elite. With P_s as the sorted population, the elite part P_e is generated by the following steps:

- While $|P_e| < E$, do:
  1) If $|P_d| < E_d$, then:
     a) If exists $\vec{x_i} \in P_s$ that can be inserted in P_d, then select $\vec{x_i}$ with lowest index i.
     b) Else, from P_s select $\vec{x_i}$ with greatest Δ
     c) Insert selected $\vec{x_i}$ in P_d and P_e
     d) Remove selected $\vec{x_i}$ from P_s
  2) Else:
     a) From P_s select $\vec{x_i}$ with lowest index i
     b) Insert selected $\vec{x_i}$ in P_e
     c) Remove selected $\vec{x_i}$ from P_s

The parameter control is performed by dividing the optimization procedure into two phases: exploration and exploitation. The algorithm generates diversified solutions in the exploration phase to search the entire solution space. The algorithm increases the evolutionary pressure in the exploitation phase by favoring the best solutions found. By properly controlling the algorithm parameters, it is possible to balance global and local searches.

To balance between exploration and exploitation, each phase runs for $N_{it}/2$ iterations. For both phases, the parameters p_e and p_m are initially set to 0.25. With these values, half of the population is composed of offspring. With $p_e = 0.25$, it also means that each elite individual should on average generate two offspring each generation.

This initial value was defined empirically. The following initial values and ranges for each parameter were also defined empirically. Although these values may not always be optimal, they are reasonable and straightforward enough to be used as base values.

The exploration phase initializes the optimization process. The diversification parameters p_d and δ are used to explore the search space. The diversification fractions starts as $p_d = 0$, and is modified in steps of $k = 0.01$. This parameter is updated during the exploration phase to keep the population diversity $Div(P)$ above δ. If the current $Div(P) < \delta$, p_d is increased by k. Otherwise, p_d is decreased by k. The parameter p_d is kept in the range $[0, 0.5]$, diversifying at most half of the elite part.

If the $Div(P)$ is still below δ when $p_d = 0.5$, then the parameter p_m is also modified. The parameter is kept in the range $[0.25, 0.5]$ and is updated using step k. By increasing the random part of the population, the diversity of the population is increased. At the maximum value of $p_m = 0.5$, half of the population is uniformly randomly generated each iteration. Also, only $1/4$ of the population is offspring, with each elite individual generating on average one offspring and each non-elite parent being on average $2/3$ of the time a random individual.

The exploration diversity threshold δ is used to define the diversity of the exploration phase. The user can define it as an algorithm parameter. However, the value 0.4 should be reasonable, in general, and is used as the default value for this parameter. This value was selected due to the nature of uniformly randomly generated solutions and global search.

Considering this, δ is set to 0.4 by default. With this value, the population of the exploration phase has the property of being similar to a uniform distribution without losing the generation of reasonable solutions. Although, in general, it should not be necessary to change this value, in some cases, it can be interesting to lower the parameter to limit the exploration.

In the exploitation phase, the algorithm eliminates the diversity parameters. The parameter p_m is reset to the initial value 0.25 if modified during the exploration. The only parameter modified during this phase is the elite fraction p_e.

The algorithm forces the search to converge to the best solution found during this phase. To increase the convergence, the parameter p_e is increased in the range $[0.25, 0.5]$, using step k. At the maximum value of $p_e = 0.5$, half of the population is composed of elite solutions, and each elite individual will generate, on average, one offspring.

Considering the modifications proposed, the parameters p_e, p_m and c_{pr} are no longer user-defined. The only parameters that still need to be defined are the number of iterations N_{it} and the population size p. The exploration diversity threshold δ can also be defined if necessary.

This proposed parameter control aims to find a reasonable balance between usability complexity and optimization efficiency. The proposed control does not guarantee the selection of optimal values for every problem. However, it should select reasonable values for any general optimization, as it uses the generic concept of exploration and exploitation. Although the algorithm may not execute the most optimal search for some specific problem, the decreased number of parameters allows the user to focus more on the problem modeling and less on the calibration of the algorithm.

B. Parallelism

To increase the scalability of the algorithm, a master-slave model was developed using CPU threads. During fitness evaluation, the master thread divides the population into equal-sized chunks and distributes them to other threads. More formally, considering a number t of threads (e.g., the number of processors), the master will divide the population into chunks of size p/t. Each thread will evaluate the fitness of individuals in its chunk, effectively reducing the processing time of fitness evaluation, which is usually the most time-consuming step of the optimizer.

C. Codification and fitness

The fitness for the multi-objective model of the PSP is a tuple (F_1, F_2, F_3), where F_1, F_2, and F_3 are the objectives defined by the Expressions (1)-(3), respectively. The decoder function is a function that will receive a coded solution $x = (x_1, x_2, x_3, ..., x_n)$, where n is the size of an encoded chromosome, which is problem-specific, and each x_i is in the domain $[0, 1]$. The decoder should map this chromosome into a problem-specific solution, which will then be evaluated by the fitness function.

In this work, two decoders are used in two different algorithm executions. The first decoder, named fragment decoder, takes an $F = L/9$-sized chromosome, with L being the amino acid sequence length, and maps it into an angle list by inserting fragments of size nine. Each of these fragments is a continuous sequence of nine amino acids extracted from some known protein structure.

For each amino acid in the protein to be predicted, 200 fragments of size nine were generated using the Robetta server [4]. These fragments were predicted excluding homologous proteins. As the fragments are size nine, each selected fragment contributes nine pairs of ϕ and ψ angles in the solution. The torsion angles ϕ and ψ are extracted from each fragment and used to build the solution.

In the fragment decoder, each variable of the chromosome x (candidate solution) is used to select a fragment from the list of 200 fragments of an amino acid. The x_i variable represents a fragment f_i that starts on the amino acid with position $p_i = 9 \times (i-1)+1$ in the protein sequence. The inserted fragment f_i is the one with position $\lceil 200 \times x_i \rceil$ in the fragment list of amino acid p_i. Figure 3a shows the decoding process.

The second decoder, named residue decoder, takes an L-sized chromosome y and maps it into an angle list by extracting the ϕ and ψ angles from each variable. The y_i variable is mapped into torsion angles ϕ_i and ψ_i by using 14 digits as scale, where the first 7 digits are used to generate ϕ_i and the remaining 7 are used to generate ψ_i. If $D_i = \{d_1, d_2, ..., d_7\}$ are the 7 digits used to generate ϕ_i, the angle ϕ_i can be defined as $\phi_i = -180 + 360 \times r$ where $r = 0.d_1 d_2 ... d_7$ (the ψ angle is defined similarly). Figure 3b exemplifies this decoding process.

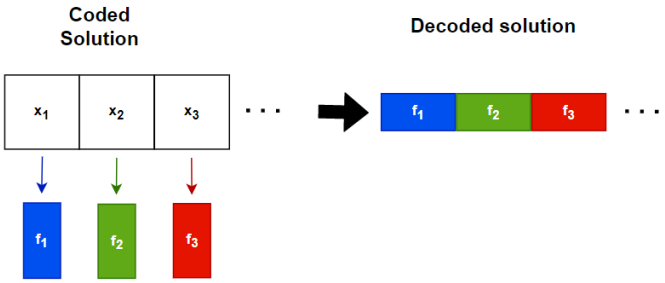
D. Predictor

The predictor first applies the MOBO with the fragment decoder, MOBO/FRAG, to predict protein structures. This algorithm will search the structure space using fragments, generating valid low-resolution structures. Then, the MOBO with the residue decoder, MOBO/RES, optimizes the initial archive of solutions.

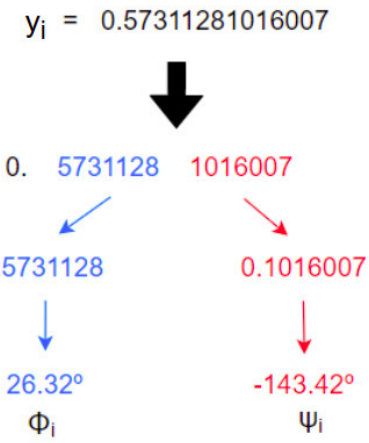
Using the archive from MOBO/FRAG as the initial population, the MOBO/RES can refine the results and increase their resolution. The archive returned by the MOBO/RES is the predicted set of protein structures. Then, a decision-making step is applied to select the final predicted protein structure. To accomplish that, the MUFOLD-CL method [30] is used to cluster the final set.

The MUFOLD-CL works by first estimating potential cluster representatives (center of a cluster) using a structural difference metric. These representatives are then used to cluster the remaining structures, also using this metric. Finally, after all clusters are formed, new representatives are selected for each cluster using a structural similarity metric that is able to describe more accurately the center of a cluster [30]. The representative structure of the largest group is returned as the predicted structure of the proposed method.

[4]http://old.robetta.org/



(a) Fragment decoder.



(b) Residue decoder.

Fig. 3: MOBO decoders

A diagram with a visualization of the complete predictor can be seen in Figure 4. In this diagram, the primary input is the amino acid sequence. The secondary structure, contact map, and fragment information are generated from this sequence. This information is used to feed the optimizer, which starts with MOBO/FRAG method. This method generates an archive of solutions, used as the initial population for the second method, MOBO/RES. The output of the second method is the optimizer output. This output is also an archive of solutions, used as input for the clustering method MUFOLD-CL. The final structure is then selected from the largest cluster found.

V. Experiments, Results & Analysis

Table I shows the protein set employed in the experiments. All proteins were taken from the RCSB database [5], with the exception of proteins T0868, T0900, T0968s1, and T1010, which were taken from the CASP competition [6]. An amount of 20 proteins with different sizes and different types of secondary structures were selected.

The Root Mean Square Deviation (RMSD) metric was utilized to measure the quality of the predicted structures. The RMSD measures the distance of atoms between two superimposed structures, with lower values indicating higher similarity structures [31]. The distance is taken considering

[5]https://www.rcsb.org/
[6]https://predictioncenter.org/

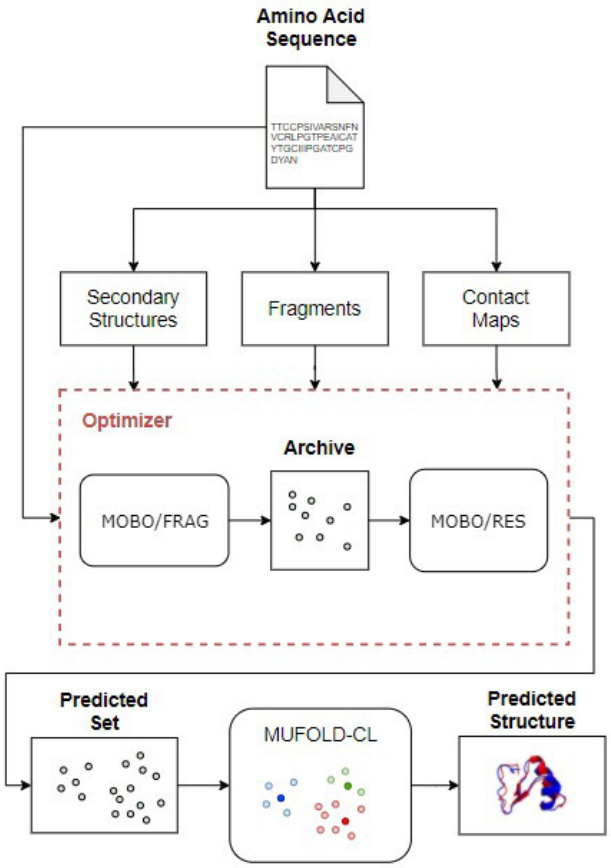


Fig. 4: Predictor

Protein	Size	Class	Protein	Size	Class
1ACW	29	$\alpha\beta$	2KDL	56	α
1DFN	30	β	1BDD	60	α
1ZDD	34	α	1ROP	63	α
1I6C	39	β	1AIL	73	α
2MR9	44	α	1HHP	99	β
2P81	44	α	T0900	106	β
1AB1	46	$\alpha\beta$	T0968s1	119	$\alpha\beta$
1CRN	46	$\alpha\beta$	1ALY	146	β
1ENH	54	α	T0868	161	α
1GB1	56	$\alpha\beta$	T1010	210	β

TABLE I: Proteins utilized in the experiments

the C_α atoms (protein backbone) and can be mathematically defined as:

$$RMSD(\vec{a},\vec{b}) = \sqrt{\frac{\sum_{i=1}^{L} d(a_i,b_i)^2}{L}} \quad (10)$$

where L is the number of residues (amino acids), $\vec{a}$ and $\vec{b}$ are two superimposed structures, and $d(a_i,b_i)$ returns the Euclidean distance (in angstroms) between the atoms a_i and b_i. The best and average values (with standard deviation) were measured for all proteins.

All the experiments were executed 20 times for statistical validation. The proposed predictor was implemented using C++17, and experiments were performed on an Ubuntu 18.04 system with an Intel Xeon E7-8860 @ 80x 2.26GHz and 1TB RAM. The utilized system has a NUMA architecture [32] with four nodes, each node with 20 cores (10 physical and 10 virtual).

The experiments were conducted using the number of iterations $N_{it} = 1000$ and the population size $p = 500$. Both MOBO/FRAG and MOBO/RES use the same values, resulting in 1,000,000 fitness evaluations. However, the MOBO/RES also defines the exploration diversity threshold as $\delta = 0.25$. This definition occurs to limit the exploration of new solutions in the MOBO/RES, whose objective is to refine the solutions found by the MOBO/FRAG. These parameters were empirically defined.

A. Predictor performance analysis

The first experiment compared the results obtained by MOBO against the previous work [12]. The previous algorithm has the same structure as MOBO, including the diversity modifications, but uses static parameters defined before the algorithm execution. The ANOVA test with 95% confidence interval was performed to validate the statistical difference between the result of the MOBO and the previous algorithm. From the results obtained, both algorithms obtained statistically same results in all 20 proteins. As one objective of using online parameter control strategies is to reduce the usability complexity of the algorithm, the MOBO algorithm has an advantage over the previous work with fewer parameters to be adjusted. Therefore, the use of parameter control is validated reducing the number of parameters to be tuned from 6 to 2. The only parameters that still need to be defined are the number of iterations N_{it} and the population size p.

In the following, three types of results are analyzed considering the proposed MOBO predictor:

- Best value of the final archive: MOBO$_{BEST}$;
- Average value of the final archive: MOBO$_{MEAN}$; and
- Value from the structure selected with MUFOLD-CL: MOBO$_{CL}$.

This analysis was performed to verify the overall quality of generated archive and the quality of the solution selected with MUFOLD-CL. Although it is not expected that the decision-maker will always select the best-generated structure, it should at least select a structure with quality higher than the average generated structure.

Comparisons with predictors from the literature were also conducted, which can be seen in Table II. For the Rosetta *de novo* protocol, the set of proteins was predicted locally. The results of the other predictors were extracted directly from their respective works.

The predictor proposed in [21] uses the MUFOLD-CL to select a final solution from the optimized set of structures. In [17], a hierarchical clustering method is used for this purpose. In [18], a single final solution was not selected, using the best solution generated by the algorithms for evaluation instead.

The overall results are shown in Table III. The table compares each protein with the results of all methods. The best and average values from other works were utilized, if available. Missing results are marked with '-' in the table.

The ANOVA test with 95% confidence interval was performed to validate the statistical difference between the result of the proposed predictor MOBO (with MUFOLD-CL) and the other predictors. At the end of Table III, an extra row (B/S/W) was added to summarize the result of this test. B represents the number of proteins where the competing method was statistically better than the proposed predictor, and the W is the number of times where the competing method was statistically worse than the proposed predictor. S is the number of proteins where there was no statistical difference between the results of the competing method and the proposed predictor.

TABLE II: Compared methods

Reference	Algorithm	Function evaluations
[21]	MODE-K	100,000 (10^5)
[18]	NSGA-II GDE3 DEMO	1,000,000 (10^6)
[17]	MOPSO	100,000 (10^5)
Rosetta	*de novo* protocol	1,000,000 (10^6)
Proposal	**MOBO**	**1,000,000** (10^6)

Considering the best solution found on all executions, the MOBO was able to generate the best solution for almost all proteins except for 1ACW (Rosetta), 1GB1 (Rosetta), 1ZDD (DEMO), and 2P81 (MODE-K). The quality of the solutions found using MUFOLD-CL was not far from the best solution generated by the MOBO. It seems that, on average, the distance between the best solution and the MUFOLD-CL solution is between 1 and 3 Å.

Overall, the solutions selected by the MUFOLD-CL are better than the other methods. The MOBO with MUFOLD-CL selected better solutions most of the time when compared with NSGA2, GDE3, DEMO, and Rosetta.

Considering the best and mean values of the MOBO, the MUFOLD-CL was always worst than the best and almost always similar to the mean. These results make sense, as the MUFOLD-CL is a clustering method and selects as the final result the center of the largest cluster. As this center is similar to all the other structures of the cluster, and the largest cluster is selected, it is expected that the results are close to the mean of the final frontier.

1) Predicted structures visualization: The visualization of the predicted structures can be seen in Figure 5. In this figure, the best solution predicted by the MOBO with MUFOLD-CL is in red, while the native structure is in blue. The structures were overlapped using the α-carbons. This overlap displays the quality of the predicted structures.

It is possible to see that the α-helices of the proteins were accurately predicted, in general. This behavior can be seen in the structures of 1AB1, 1BDD, 1CRN, 1ENH, 1GB1, 1ROP, 1ZDD, and 2MR9. However, for some α proteins, the algorithm was unable to generate accurate helices. This can be observed in 2KDL and 2P81, which are small α proteins with poor predictions. This divergence of quality is probably due to the secondary structure and contact map information predicted for these proteins. If the input information has poor quality, it is expected that the output structure will be equally bad.

It is also possible to see in this visualization that one difficult part is the prediction of β-sheets. These structures are harder to predict than the α-helices, and proteins of the class β are usually the ones with the lowest prediction quality. This difficulty can be observed both in simpler proteins, such as 1DFN, and in the more complex, such as 1ALY and 1HHP.

VI. Conclusion and Future Work

The PSP problem is one of the most important open problem in biology. As proteins are a core biological macromolecule, understanding their structure and functionality is essential to understanding complex biological processes. For this purpose, computational methods are employed to build and simulate protein structures.

Among possible methods to approach the PSP problem, multi-objective optimization shows great potential. Being able to simultaneously optimize multiple conflicting objectives, these methods can optimize complex mathematical models. Due to this, recent works in the literature have invested multi-objective optimization for the PSP problem.

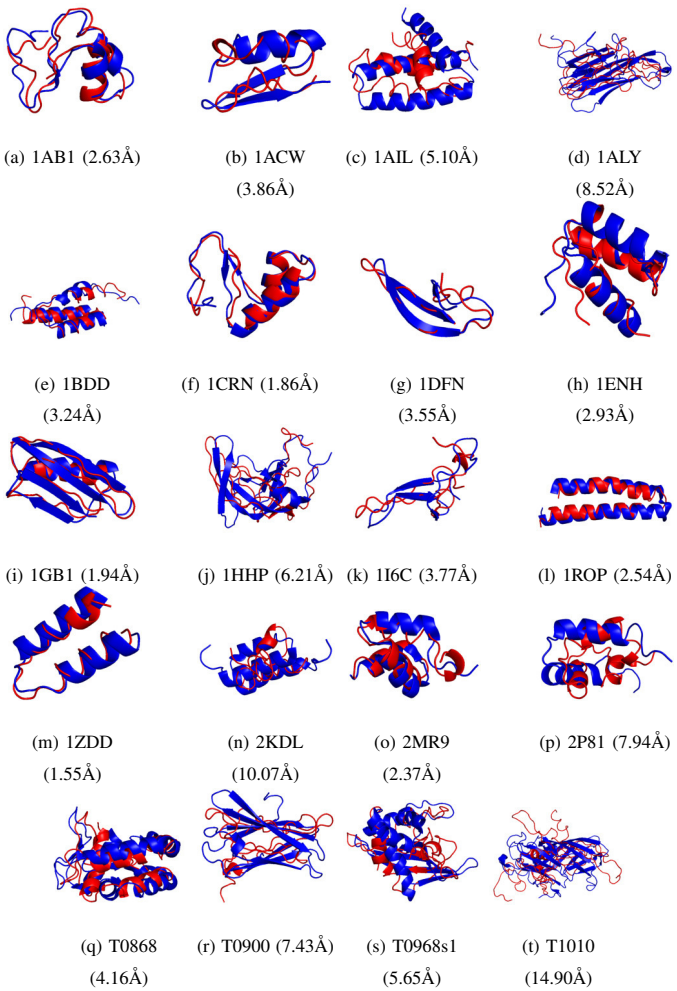


Fig. 5: Overlap of predicted and native figures. The predicted structure is in red and the native structure is in blue.

TABLE III: RMSD results. For each method, the best solution found ($f*$), mean value ($\overline{x}$) and standard deviation (s) are displayed. The best absolute $f*$ for each protein is set in bold.

Protein		NSGA2	GDE3	DEMO	MOPSO	MODE-K	Rosetta	MOBO$_{BEST}$	MOBO$_{MEAN}$	MOBO$_{CL}$
1AB1	$f*$	-	-	-	9.80	7.38	4.90	**2.17**	4.72	2.63
	$\overline{x}$	-	-	-	-	-	7.78	2.93	5.81	5.08
	s	-	-	-	-	-	1.20	0.46	0.75	1.61
1ACW	$f*$	3.81	3.63	3.82	-	-	**1.65**	2.71	4.47	3.86
	$\overline{x}$	6.47	6.56	7.17	-	-	5.48	3.90	5.55	5.29
	s	1.58	1.72	1.81	-	-	1.27	0.58	0.44	0.55
1AIL	$f*$	7.07	3.25	**3.14**	-	-	6.26	3.84	5.96	5.10
	$\overline{x}$	10.30	6.77	7.40	-	-	9.50	5.72	7.78	7.42
	s	1.38	2.81	2.55	-	-	1.98	0.89	0.81	1.31
1ALY	$f*$	-	-	-	-	-	13.62	**7.78**	14.29	8.52
	$\overline{x}$	-	-	-	-	-	16.51	11.29	17.29	14.59
	s	-	-	-	-	-	2.14	1.75	1.29	2.93
1BDD	$f*$	-	-	-	5.64	4.98	5.27	**2.79**	3.76	3.24
	$\overline{x}$	-	-	-	-	-	7.61	3.26	4.15	3.94
	s	-	-	-	-	-	1.60	0.19	0.29	0.59
1CRN	$f*$	6.23	5.13	6.32	7.57	-	4.91	**1.65**	3.53	1.86
	$\overline{x}$	9.24	9.62	9.31	-	-	7.24	2.52	5.28	4.72
	s	1.50	2.69	2.79	-	-	1.02	0.45	1.04	2.08
1DFN	$f*$	-	-	-	-	7.00	5.48	**2.68**	4.49	3.55
	$\overline{x}$	-	-	-	-	-	7.10	3.23	5.12	4.72
	s	-	-	-	-	-	0.68	0.34	0.57	1.05
1ENH	$f*$	6.84	3.10	4.29	8.92	7.80	3.66	**1.94**	3.29	2.93
	$\overline{x}$	10.14	8.09	7.47	-	-	5.00	2.30	3.86	3.67
	s	1.31	2.79	1.67	-	-	1.03	0.26	0.33	0.62
1GB1	$f*$	-	-	-	-	-	**1.63**	**1.63**	2.53	1.94
	$\overline{x}$	-	-	-	-	-	2.92	2.45	3.38	2.94
	s	-	-	-	-	-	1.82	0.53	0.63	0.74
1HHP	$f*$	-	-	-	-	-	11.32	**4.28**	8.83	6.21
	$\overline{x}$	-	-	-	-	-	14.58	8.57	12.80	10.55
	s	-	-	-	-	-	1.28	2.26	1.93	2.63
1I6C	$f*$	-	-	-	8.47	7.76	6.84	**3.21**	4.66	3.77
	$\overline{x}$	-	-	-	-	-	8.31	3.74	5.37	5.32
	s	-	-	-	-	-	0.76	0.41	0.35	0.76
1ROP	$f*$	5.96	2.74	3.04	3.51	3.01	8.34	**1.14**	2.56	2.54
	$\overline{x}$	10.86	7.05	6.80	-	-	11.01	1.63	3.56	3.18
	s	1.85	2.07	1.61	-	-	1.52	0.43	0.99	0.52
1ZDD	$f*$	3.47	1.79	**1.19**	2.15	2.50	2.99	1.31	1.54	1.55
	$\overline{x}$	6.04	4.05	4.01	-	-	5.26	1.58	1.83	1.83
	s	1.29	1.16	1.98	-	-	1.06	0.18	0.22	0.21
2KDL	$f*$	-	-	-	10.29	7.72	11.41	**6.01**	10.57	10.07
	$\overline{x}$	-	-	-	-	-	12.98	8.55	11.26	11.13
	s	-	-	-	-	-	0.44	0.97	0.37	0.67
2MR9	$f*$	6.16	3.00	2.62	-	-	2.21	**1.66**	2.41	2.37
	$\overline{x}$	8.29	7.09	7.21	-	-	4.46	1.89	2.61	2.61
	s	1.29	1.87	1.87	-	-	2.62	0.20	0.12	0.18
2P81	$f*$	5.85	4.78	5.06	6.28	**4.76**	5.56	5.89	8.33	7.94
	$\overline{x}$	8.94	7.10	7.72	-	-	7.97	7.12	9.08	8.94
	s	1.30	1.34	1.44	-	-	1.34	0.67	0.37	0.49
T0868	$f*$	-	-	-	-	-	13.06	**3.38**	6.00	4.16
	$\overline{x}$	-	-	-	-	-	15.26	5.20	8.82	6.90
	s	-	-	-	-	-	1.41	1.15	1.46	1.63
T0900	$f*$	-	-	-	-	-	12.71	**5.93**	10.72	7.43
	$\overline{x}$	-	-	-	-	-	15.07	7.39	12.21	9.40
	s	-	-	-	-	-	0.88	0.74	0.66	1.33
T0968S1	$f*$	-	-	-	-	-	12.79	**4.85**	8.47	5.65
	$\overline{x}$	-	-	-	-	-	15.78	6.50	11.43	8.93
	s	-	-	-	-	-	1.67	1.02	1.49	2.00
T1010	$f*$	-	-	-	-	-	18.37	**12.86**	20.68	14.90
	$\overline{x}$	-	-	-	-	-	21.13	14.36	22.50	19.46
	s	-	-	-	-	-	1.43	0.88	0.97	3.53
B/S/W		0/1/7	1/1/6	1/1/6	-	-	1/3/16	20/0/0	0/14/6	-

The results obtained by the proposed approach demonstrated that the predictor with the decision-making step is highly competitive with other works from the literature. Although the predictor is consistently able to predict α-helices, the prediction of β-sheet is unstable, with proteins of the β class having considerably lower quality when compared with proteins of the α class.

It is possible to define some improvements in the proposed

work. Multiple predictors could be used to reduce the inaccuracy of predicted information in the optimization. By using the information of different predictors, it is possible the reduce bias and combine the best information from each source. This combination of multiple data sources may increase the effectiveness of secondary structures and contact maps in the prediction of tertiary structures.

Also, the prediction of β should be more thoroughly researched. Other types of information could be added to the optimization model with the objective of increasing the quality of β-sheets in protein predictions. By focusing on this weak point, the overall effectiveness of the predictor should increase not only for β proteins but also for all proteins in general.

Another line of work is to further improve the search algorithm itself. This work presents a simple online parameter control that is able to select reasonable values. However, more complex techniques could be employed, such as the use of fuzzy systems to incorporate expert information about the problem that is being optimized. Also, local search could be implemented to further specialize the proposed algorithm for the optimization of protein structures.

REFERENCES

[1] R. Garret and C. Grisham, "Biochemistry 4ed," *University of Virginia, Boston, MA*, 2010.

[2] K. A. Dill, S. B. Ozkan, M. S. Shell, and T. R. Weikl, "The protein folding problem," *Annu. Rev. Biophys.*, vol. 37, pp. 289–316, 2008.

[3] J. Gu and P. E. Bourne, *Structural bioinformatics*. Hoboken: John Wiley & Sons, 2009, vol. 44.

[4] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko *et al.*, "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[5] A. David, S. Islam, E. Tankhilevich, and M. J. Sternberg, "The alphafold database of protein structures: A biologist's guide," *Journal of Molecular Biology*, vol. 434, no. 2, p. 167336, 2022.

[6] M. Dorn, M. B. e Silva, L. S. Buriol, and L. C. Lamb, "Three-dimensional protein structure prediction: Methods and computational strategies," *Computational biology and chemistry*, vol. 53, pp. 251–276, 2014.

[7] A. E. Márquez-Chamorro, G. Asencio-Cortés, C. E. Santiesteban-Toca, and J. S. Aguilar-Ruiz, "Soft computing methods for the prediction of protein tertiary structures: A survey," *Applied Soft Computing*, vol. 35, pp. 398–410, 2015.

[8] R. S. Silva and R. S. Parpinelli, "A self-adaptive differential evolution with fragment insertion for the protein structure prediction problem," in *International Workshop on Hybrid Metaheuristics*. Springer, 2019, pp. 136–149.

[9] N. N. Will and R. S. Parpinelli, "Comparing best and quota fragment picker protocols applied to protein structure prediction." in *HIS*, 2020, pp. 669–678.

[10] V. Cutello, G. Narzisi, and G. Nicosia, "A multi-objective evolutionary approach to the protein structure prediction problem," *Journal of The Royal Society Interface*, vol. 3, no. 6, pp. 139–151, 2006.

[11] D. Kalyanmoy and K. Deb, "Multi-objective optimization using evolutionary algorithms," *West Sussex, England: John Wiley*, 2001.

[12] F. Marchi and R. S. Parpinelli, "A multi-objective approach to the protein structure prediction problem using the biased random-key genetic algorithm," in *2021 IEEE Congress on Evolutionary Computation (CEC)*. New York: IEEE, 2021, pp. 1070–1077.

[13] J. F. Gonçalves and M. G. Resende, "Biased random-key genetic algorithms for combinatorial optimization," *Journal of Heuristics*, vol. 17, no. 5, pp. 487–525, 2011.

[14] S. M. Venske, R. A. Gonçalves, E. M. Benelli, and M. R. Delgado, "Adem o/d: An adaptive differential evolution for protein structure prediction problem," *Expert Systems with Applications*, vol. 56, pp. 209–226, 2016.

[15] S. Gao, S. Song, J. Cheng, Y. Todo, and M. Zhou, "Incorporation of solvent effect into multi-objective evolutionary algorithm for improved protein structure prediction," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 15, no. 4, pp. 1365–1378, 2017.

[16] S. Song, S. Gao, X. Chen, D. Jia, X. Qian, and Y. Todo, "Aimoes: Archive information assisted multi-objective evolutionary strategy for ab initio protein structure prediction," *Knowledge-Based Systems*, vol. 146, pp. 58–72, 2018.

[17] S. Song, J. Ji, X. Chen, S. Gao, Z. Tang, and Y. Todo, "Adoption of an improved pso to explore a compound multi-objective energy function in protein structure prediction," *Applied Soft Computing*, vol. 72, pp. 539–551, 2018.

[18] P. H. Narloch, M. J. Krause, and M. Dorn, "Multi-objective differential evolution algorithms for the protein structure prediction problem," in *2020 IEEE Congress on Evolutionary Computation (CEC)*. New York: IEEE, 2020, pp. 1–8.

[19] G. K. Rocha, K. B. Dos Santos, J. S. Angelo, F. L. Custodio, H. J. Barbosa, and L. E. Dardenne, "Inserting co-evolution information from contact maps into a multiobjective genetic algorithm for protein structure prediction," in *2018 IEEE Congress on Evolutionary Computation (CEC)*. New York: IEEE, 2018, pp. 1–8.

[20] A. B. Zaman, P. V. Parthasarathy, and A. Shehu, "Using sequence-predicted contacts to guide template-free protein structure prediction," in *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, ser. BCB '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 154–160. [Online]. Available: https://doi.org/10.1145/3307339.3342175

[21] X. Chen, S. Song, J. Ji, Z. Tang, and Y. Todo, "Incorporating a multiobjective knowledge-based energy function into differential evolution for protein structure prediction," *Information Sciences*, vol. 540, pp. 69–88, 2020.

[22] L. de Lima Corrêa and M. Dorn, "A multi-objective swarm-based algorithm for the prediction of protein structures," in *International Conference on Computational Science*. Cham: Springer, 2019, pp. 101–115.

[23] J. C. C. Tudela and J. O. Lopera, "Parallel protein structure prediction by multiobjective optimization," in *2009 17th Euromicro International Conference on Parallel, Distributed and Network-based Processing*. New York: IEEE, 2009, pp. 268–275.

[24] C. A. Rohl, C. E. Strauss, K. M. Misura, and D. Baker, "Protein structure prediction using rosetta," *Methods in enzymology*, vol. 383, pp. 66–93, 2004.

[25] D. T. Jones, "Protein secondary structure prediction based on position-specific scoring matrices," *Journal of molecular biology*, vol. 292, no. 2, pp. 195–202, 1999.

[26] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features," *Biopolymers: Original Research on Biomolecules*, vol. 22, no. 12, pp. 2577–2637, 1983.

[27] J. Ma, S. Wang, Z. Wang, and J. Xu, "Protein contact prediction by integrating joint evolutionary coupling analysis and supervised learning," *Bioinformatics*, vol. 31, no. 21, pp. 3506–3513, 2015.

[28] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[29] R. Parpinelli, G. Plichoski, R. Silva, and P. Narloch, "A review of techniques for online control of parameters in swarm intelligence and evolutionary computation algorithms," *International Journal of Bio-Inspired Computation*, vol. 13, pp. 1–20, 2019.

[30] J. Zhang and D. Xu, "Fast algorithm for population-based protein structural model analysis," *Proteomics*, vol. 13, no. 2, pp. 221–229, 2013.

[31] V. N. Maiorov and G. M. Crippen, "Significance of root-mean-square deviation in comparing three-dimensional structures of globular proteins," *Journal of molecular biology*, vol. 235, no. 2, pp. 625–634, 1994.

[32] H. El-Rewini and M. Abd-El-Barr, *Advanced computer architecture and parallel processing*. Hoboken, New Jersey: John Wiley & Sons, 2005, vol. 42.