Vladimir FILIPOVIĆ*

METAHEURISTIC OPTIMIZATIONS IN BIOINFORMATICS AND MACHINE LEARNING

Abstract: This paper deals with the usage of metaheuristic optimization methods aimed at solving problems in bioinformatics and machine learning. Two metaheuristic optimization methods have been selected and explained in detail: Electromagnetism-like Metaheuristics and Variable Neighborhood Search. Results obtained by applying those two metaheuristic optimization methods on various problems in bioinformatics and machine learning are described. More precisely, the following problems have been solved: dimensionality reduction, support vector machine parameter selection, maximum betweeness problem and k-plex partitioning problem. At the end, algebraic topology concepts aimed at enhancing Electromagnetism-like Metaheuristics and Variable Neighborhood Search are proposed and their design is described.

Key words: *Metaheuristics, Optimization, Electromagnetism-like Metaheuristics, Variable Neighborhood Search, Algebraic topology*

1. INTRODUCTION

In recent years, there has been a tremendous growth of interest in applications of optimization in bioinformatics and machine learning. Optimization is frequently used for designing and modeling complex systems, which are essential in biomedical and biological research, e. g. in solving the maximum betweeness problem [1], in partitioning complex networks to k-plex structures [2], in identifying functionally related protein groups in weighted PPI networks [3], etc.

The main purpose of this paper is to address the usage of metaheuristic optimization methods in bioinformatics and in machine learning. In order to do so, we will concentrate on three problems from these domains (Dimensionality Reduction Problem, Maximum Betweeness Problem and Maximum Edge k-plex Partitioning Problem) and on two metaheuristic

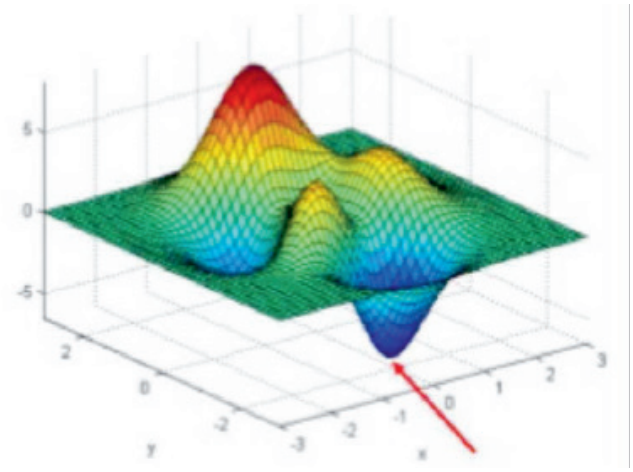
* Vladimir Filipović, University of Belgrade, Faculty of Mathematics, Department of Computer Science, Belgrade, Serbia

optimization methods (Electromagnetism-like Metaheuristics and Variable Neighborhood Search).

2. PROBLEMS

Firstly, let us define the optimization problems. Since minimization and maximization optimization problems have basically the same structure, the minimization optimization is defined.

Following elements are known: search space S; solution space X, $X \subseteq S$; objective function f, $f\colon S \longrightarrow R$, which maps elements of S to real numbers. In minimization optimization problems, the goal is to calculate $x^* \in X$, such that $f(x^*) = min\{f(x) \mid x \in X\}$. Picture 1 illustrates one example of the minimization optimization problem.



Picture 1 — Minimization optimization problem illustration

There are many important optimization problems in the fields of machine learning and bioinformatics, the most relevant are enlisted in [4] and [5].

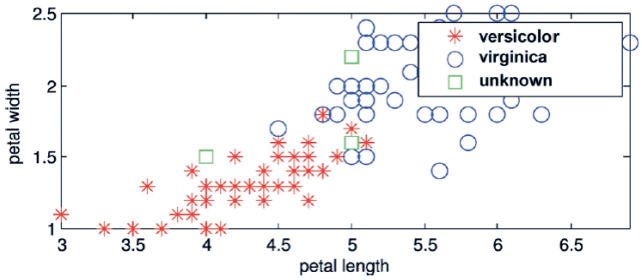
This paper will address three machine learning and bioinformatics combinatorial optimization problems: Dimensionality Reduction Problem, Maximum Betweeness Problem and Edge-weight k-plex Partition Problem.

2.1. DIMENSIONALITY REDUCTION PROBLEM

Data mining is one of the most popular and exciting discipline of applied informatics. It allows researchers to discover complex and hidden patterns in data, which can potentially lead to completely new conclusions in different disciplines, where sometimes even experts in the disciplines

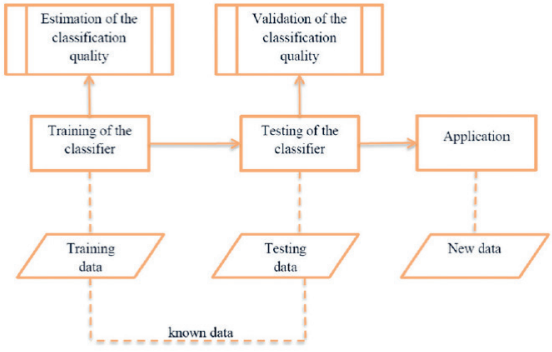
cannot do better. Nowadays, there is an extremely rapid growth in the volume of data stored in biological databases, with increased complexity of data and a very high dimensionality.

Data mining includes classification, which predicts a certain outcome based on a given input. An illustrative example of a classification task is given in Picture 2.



Picture 2 — Basic concept of classification

In order to learn how to predict outcome, the algorithm uses a set of training records containing a set of attributes and the respective outcome. The classification algorithm then, in the so-called training phase, tries to discover relationships between the attributes that would make it possible to predict an outcome. After this step, the algorithm is given a dataset not seen before, called a set of testing records, which contains the same set of attributes, except for the prediction attribute that is not yet known. The algorithm analyses the input and produces a prediction — this is the testing phase. The prediction accuracy defines the quality of the classification algorithm. After the testing phase, classifier is used in real-life conditions. The classification process is described by the flowchart in Picture 3.



Picture 3 — Flowchart of the classification process

There are two types of benefits for applying dimensionality reduction by feature selection for the classification process: firstly, by eliminating unnecessary features, it is possible to eliminate dataset noise that degrades the quality of the classification model; secondly, the problem dimension is decreased and the efficiency is increased.

The criteria for dimensionality reduction can vary, and in terms of classification problems (which are one focus of this paper) are usually referred to the classification accuracy, model efficiency, level of dimension reduction, or composition of former criteria. Dimensionality reduction algorithms have been utilized as a support for solving various and often real problems. There are three general types of dimensionality reduction algorithms [6]:

1) wrapper methods that use a classification algorithm as a black box for the evaluation of feature subsets;

2) filter methods that form feature subsets in the preprocessing phase, and do not depend on the employed classification algorithm;

3) embedded methods that form a feature subset in the training process and are specific to a given classification algorithm.

Various optimization techniques are used for dimensionality reduction by feature selection: genetic programming, fractional 0–1 programming, neural-genetic algorithms, particle swarm optimization, etc. In-depth study of Dimensionality Reduction Problem is given in [6], where the wrapper method is considered for dimensionality reduction, where the 1-nearest-neighbor classifier (1-NN) and support vector machine (SVM) are used as underlying classification mechanisms. Here, specific Electromagnetism-like Metaheuristics is proposed, various computational experiments are executed and obtained results are compared to results reported from other state-of-the-art methods for the considered problem.

2.2. MAXIMUM BETWEENESS PROBLEM

The betweeness problem is a well-known optimization problem. For a given finite set S of n objects $S = \{x_1, x_2, \ldots, x_n\}$ and a given set C of triples $(x_i, x_j, x_k) \in S \times S \times S$, the betweeness problem is a problem of determination of the total ordering of the elements from S, such that triples from C satisfy the „betweeness constraint“, i. e. the element x_j is between the elements x_i and x_k. The problem presented in the paper [1], called the Maximum Betweeness Problem (MBP), deals with finding the total ordering that maximizes the number of satisfied constraints.

The MBP, as well as other betweeness problems, belongs to a class of discrete optimization problems. Those problems have important ap-

plications in various fields, including bioinformatics. For example, the MBP is used for solving some physical mapping problems in molecular biology. During the radiation hybrid experiments, X-rays are used to fragment chromosomes. If the markers on chromosomes are more distant, the probability that the given dose of an X-ray will break a chromosome is greater. In this way, markers are placed on two separate chromosomal fragments. By estimating the frequency of the breaking points, and thus the distances between markers, it is possible to determine their order within a chromosome in a manner analogous to meiotic mapping. In this context, improvement of the radiation experiment can be achieved by finding the total ordering of the markers that maximizes the number of satisfied constraints.

2.3. PROBLEM OF PARTITIONING OF COMPLEX BIOLOGICAL NETWORKS (K-PLEX PARTITION PROBLEM)

Partitioning networks into high density subnetworks, especially cliques, has already been proven as a useful technique for obtaining new information in understanding complicated relations between biological elements. For example, partitioning in protein threading analysis can be reduced on maximum edge weight clique problem, the protein side chain packing problem is transformed into a problem of finding a maximum weight clique.

Finding cliques is also one of the methods for identification of the clusters that are later divided into protein complexes and dynamic functional modules. By analyzing the multibody structure of the network of protein–protein interactions (PPI), molecular modules that are densely connected within themselves, but sparsely connected with the rest of the network, are discovered. Cliques have a similar use in modular decomposition of PPI networks. This decomposition allows to combine proteins into the actual functional complexes by identifying groups of proteins acting as a single unit.

On the other hand, a number of biological networks classes contain only sparse networks. Dealing with such networks, partitioning into cliques can be too restrictive method, so many potentially useful information about the interference of biological objects can be neglected. Therefore, clique relaxation approaches could be even more useful.

In the approach presented here, partitioning is followed by the principle that the objects in each partition are still highly connected in a particular way, but not so restrictively to form a clique. By relaxing cliques to

sparse graphs, biological objects become connected in semantically or functionally logical groups which we call k-plexes, having in mind that the total sum of weights in all partitions should be as large as possible.

Here, we deal with the partitioning of the edge-weighted networks into k-plex components, where a subset of some *n*vertices in a network is a k-plex if the degree of each vertex in the subnetwork induced by this subset is at least $n - k$. The aim of the Maximum Edge-weight k-plex Partitioning Problem is to find the k-plex partitioning with the maximal total weight of edges.

More formal definition of this problem is given in [2]: Let a network be denoted as $G = (V, E)$, where $V = \{1,2, \ldots, n\}$ is the set of nodes and $E \subseteq V \times V$ is the set of edges. With uv we simply denote the edge $\{u, v\} \in E$. With real numbers $w_{uv} > 0$ we denote the weight of the edge connecting nodes u and v. We call u and v the end-vertices of the edge uv.

Let $k \geq 1$be an integer. A set of nodes S is called k-plex if the degree of each node in the subnetwork induced by S is at least $n - k$. The weight of a k-plex is the sum of all its edge weights.

The weight of the whole partition is the sum of the weights of all its k-plex components. The Maximum Edge-weight k-plex Partitioning Problem is then defined as finding such a partition of V which is of the maximum total weight and each component is a k-plex. If $k = 1$, the k-plex is a clique and the maximum edge-weight k-plex partitioning problem is brought down to the Maximum Edge-weight Clique Partitioning Problem.

3. METAHEURISTIC OPTIMIZATION METHODS

Metaheuristic methods are generalized computational intelligence methods that can be successfully adopted to various problem domains. They are trying to obtain the optimal solution, or the solution that is close to optimal one. Metaheuristic algorithms are characterized with approximation and non-determinism.

Basic metaheuristics concepts are abstractly represented. They should be adapted to problem domain, otherwise they should won't obtain enough good solution.

As the contrast to exact methods, which produces the exact solution (but have issues with time resources, with memory resources and sometimes solution cannot be obtained at all), metaheuristics methods produce approximate solution, which optimality is not guaranteed (but it works with limited computational resources, solution is always produced, and it usually has good quality).

Metaheuristic methods can be population-based (like Evolutionary algorithms, Particle Swarm Optimization, Electromagnetism-based Metaheuristics, etc.) or single-solution (like Taboo Search, Simulated Annealing, Variable Neighborhood Search etc.).

In the following subsections we will focus on two methods, one population based and one single-solution, namely to Electromagnetism-like Metaheuristic (EM) and Variable Neighborhood Search (VNS).

3.1. ELECTROMAGNETISM-LIKE METAHEURISTICS (EM)

Electromagnetism-like Metaheuristic (EM), proposed in [7], represents a population-based optimization technique inspired by mechanisms of interaction among electrically charged particles (called EM points). The method employs a proficient search process governed by EM points, where each of them represents single candidate solution of the underlying problem. EM points that represent better solutions are awarded with higher charge. This is crucial for leading the search process towards promising solution regions, because EM points with higher charge attract other points more strongly. The exact attraction-repulsion relationship is given in formula analogues to Coulomb's Law.

Electromagnetism-like algorithms turn out to be successful in solving many problems with practical and theoretical background: in [8] EM technique is adopted to solve feature selection problem, EM method for uncapacitated multiple allocation hub location problem is proposed in [9], EM method for the SVM parameter tuning [10], etc.

Overall structure of the EM algorithm is described in the flowchart in Picture 4.

```
   input: N_it, M
 1 p = createInitialPoints(M);
 2 for iter ← 1 to N_it do
 3 |   for i ← 1 to M do
 4 |   |   objFunction(p_i);
 5 |   end
 6 |   charges(p);
 7 |   forces(p);
 8 |   relocate(p);
 9 end
10 printSolution();
```

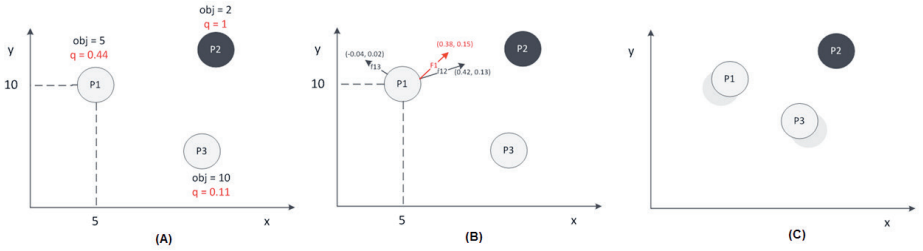
Picture 4 — Outline of the EM method

EM requires only two control parameters: $N_{it}$is the number of the main loop iterations and M represents the number of EM points. The points are first assigned with initial solutions and after that, the algorithm enters into the main loop. The main loop iterates N_{it} times and within iteration every EM point $\boldsymbol{p_i}$, $i = 1,\ldots M$ is subjected to the objective value calculation, i. e. measuring the quality of solution represented by that point.

The next step is the calculation of the EM point's charges. The charge of a fixed EM point will depend on its solution quality, according to formula:

$$q_i = e^{-N\frac{obj(\boldsymbol{p_{best}}) - obj(\boldsymbol{p_i})}{\sum_{k=1}^{M} obj(\boldsymbol{p_{best}}) - obj(\boldsymbol{p_k})}}$$

In previous formula, N is dimensionality on EM point space, obj is an objective function and $\boldsymbol{p_{best}}$ denotes EM point with the highest objective value. Illustrative example with EM points and its calculated charges is shown in Picture 5 (A).



Picture 5 — EM — (A) calculation of the charges, (B) calculation of the forces, (C) movements of the EM points

After all charges are calculated, the total impact on each point is calculated by superpositioning particle pairwise interaction forces, which are calculated by following formula:

$$\boldsymbol{F_i} = \begin{cases} \sum_{j=1, j\neq i}^{M} \left(\boldsymbol{p_j} - \boldsymbol{p_i}\right) \frac{q_j \times q_i}{\left\|\boldsymbol{p_j} - \boldsymbol{p_i}\right\|^2}, & p_j^{obj} < p_i^{obj} \\ \sum_{j=1, j\neq i}^{M} (\boldsymbol{p_i} - \boldsymbol{p_j}) \frac{q_j \times q_i}{\left\|\boldsymbol{p_j} - \boldsymbol{p_i}\right\|^2}, & p_j^{obj} \geq p_i^{obj} \end{cases}.$$

It should be noticed that forces are calculated similar to the Coulomb's Law, in a sense that the force between every two particles is proportional

to the product of their charges and inversely proportional to their distances. Picture 5 (B) shows calculated force vectors for every EM point given in previous illustrative example.

Upon calculation of all the forces $\boldsymbol{F_i} = \left(F_i^1, F_i^2, \ldots, F_i^N\right)\ i = 1,\ldots M,$ the movement procedure is applied. Movement of each EM point $\boldsymbol{p_i} = \left(p_i^1, p_i^2, \ldots, p_i^N\right)$ is guided by direction and magnitude of corresponding force vector $\boldsymbol{F_i}$. The following formula determines movements of EM points:

$$p_i^k = \begin{cases} p_i^k + \lambda \frac{F_i^k}{\|\boldsymbol{F_i}\|}\left(1 - p_i^k\right), & F_i^k > 0 \\ p_i^k + \lambda \frac{F_i^k}{\|\boldsymbol{F_i}\|} p_i^k, & F_i^k \leq 0 \end{cases}$$

Illustrative example that describes movements of EM points whose charges and forces are calculated is shown in Picture 5 (C).

Overall structure of the EM within any problem domain is like previously described. However, EMs aimed at solving different problems differs in EM point representation, in calculating objective function for EM points, and in values of the parameters used during the execution of the method.

3.2. VARIABLE NEIGHBORHOOD SEARCH (VNS)

Variable Neighborhood Search (VNS) method is a robust metaheuristic introduced by Mladenović and Hansen [11]. The main searching principle of a VNS is based on the empirical evidences: (a) multiple local optima are correlated in some sense (usually close to each other) and (b) a local optimum found in one neighborhood structure is not necessarily a local optimum for some other neighborhood structure.

The overall structure of the VNS algorithm [1] is shown on the Picture 6.

The input of the VNS algorithm, consists of:

- n_{min} and n_{max} — minimal and maximal VNS neighborhood structure size;
- it_{max}, $itrep_{max}$, t_{max} — maximal number of total iterations, maximal number of iterations without improvement, and maximal execution time in seconds, respectively;
- $prob$ — probability to move to the other solution of the same quality;
- k- size of the neighborhood (integer value);

```
input: n_min, n_max, it_max, itrep_max, t_max, prob, k
output: x
1  x ← initializeSolution();
2  n ← n_min;
3  it ← 1;
4  while it < it_max ∧ (it − it_lastimpr) < itrep_max ∧ t_run < t_max do
5  |  x′ ← shaking(x,n);
6  |  x″ ← localSearch(x′, k);
7  |  move ← shouldMove(x, x″, prob);
8  |  if move then
9  |  |  x ← x″ ;
10 |  else if n < n_max then
11 |  |  n ← n + 1;
12 |  else
13 |  |  n ← n_min;
14 |  it ← it + 1;
15 end
16 return x;
```

Picture 6 — Outline of the VNS method

The main loop VNS algorithm usually imposes three main procedures: shaking, local search (LS) and neighborhood change.

o Shaking — in order to escape local suboptimal solutions, a new solution within a parametrized neighborhood of the current best solution is generated.

o Local search — starting from the new solution obtained in the previous step, other possible solutions within local neighborhood are systematically examined with the aim of finding the local optimum.

o Neighborhood change — depending on the success of the previous two procedures, the current neighborhood size is adjusted. More precisely, when the current best solution is changed, neighborhood size is reduced to minimal, otherwise it is cyclically increased by 1 (cycle ends at maximal neighborhood size).

In the algorithm presented on Picture 6, previously described procedures are iteratively called, until no further improvements of the best solution can be made inside the current neighborhood. When that situation appears, the algorithm steps into the next neighborhood. When the last neighborhood n_{max} is explored, the search restarts at the first neighborhood n_{min}.

The execution of the VNS is stopped when either of the following conditions becomes satisfied: a maximum number of iterations is reached, a

maximum number of iterations without any improvement of the current best solution is reached, or a maximum allowed execution time is reached.

Design of the VNS for optimizing specific problem requires that solution representation, shaking procedure and LS procedure should be defined in the way that is the most efficient for that specific problem. Of course, parameters that governs VNS search process differs from problem to problem. However, overall structure of the VNS method is the one described above.

4. SOLUTIONS AND OBTAINED RESULTS

4.1. EM FOR DIMENSIONALITY REDUCTION PROBLEM

In order to achieve desired results, proposed EM metaheuristic method have the custom representation, custom shaking and custom LS. Detailed elaboration of the proposed method, computational experiments that are designed and executed, and the analysis of the obtained results (which indicate that proposed method outperforms other method on the well-known problem-instance set) is given in [6]. The same data are publicly available in the author's GitHub repository [12] in the supplemental data section.

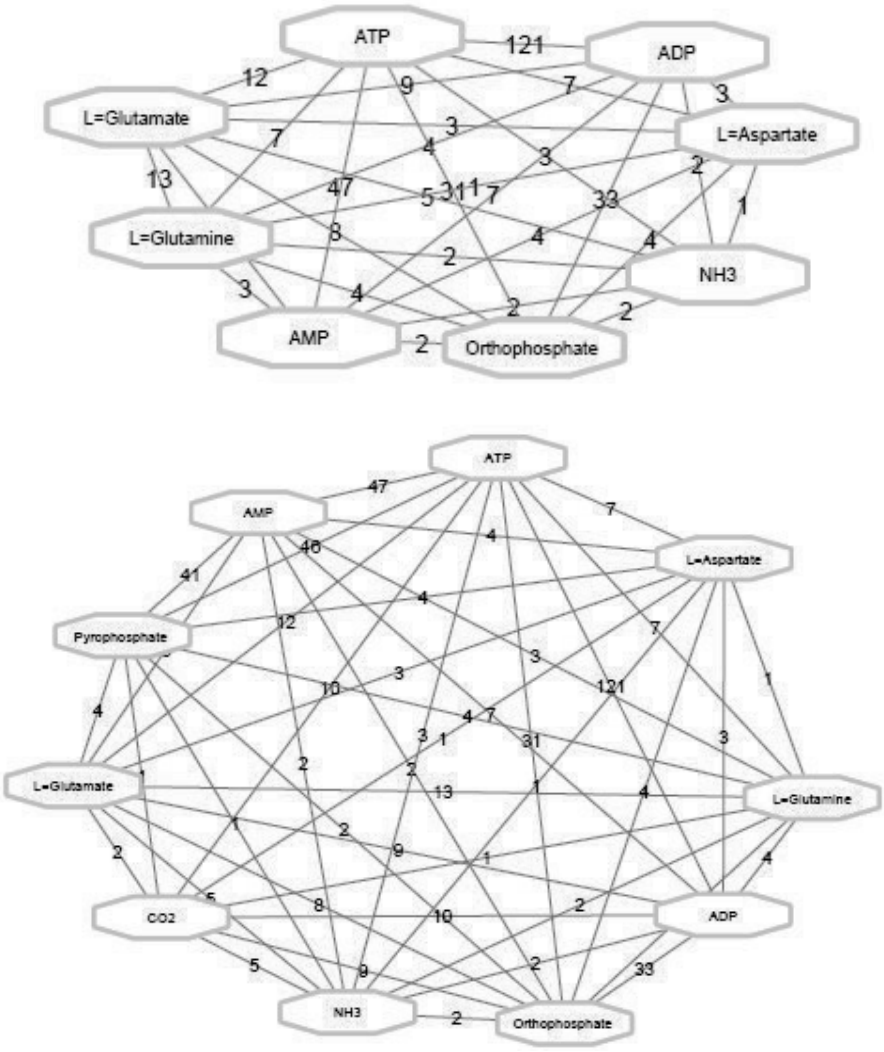
4.2. EM FOR MAXIMUM BETWEENESS PROBLEM

EM metaheuristic designed to solve Maximum Betweeness Problem have specific the EM representation and LS designed specifically for that problem. Detailed elaboration of the designed method, executed computational experiments and obtained results is given in [1]. Obtained results that indicate superiority of the designed method to other state-of-the art counterparts are publicly available as supplemental data in GitHub repository of the author [12].

4.3. VNS FOR MAXIMUM EDGE-WEIGHT K-PLEX PARTITION PROBLEM

Detailed elaboration of the proposed method specification, executed computational experiments and obtained results is given in [2]. Results from [2] clearly indicate that proposing of the newly designed method to this problem is justified.

Moreover, the relaxation of the clustering requirements lead to more useful information from biological point of view. Among many metabolic



Picture 7 — Largest clusters obtained for k=1 and k=2

processes that appeared in various k-plexes obtained by the proposed VNS algorithm, in paper [2], the following processes are discussed: amino acid degradation process, fatty acids synthesis, vitamin B6 synthesis, oxidation of the succinate to the fumarate and formaldehyde oxidation. In this section, we will set focus to amino acid degradation process — which is one of the most important processes in metabolism.

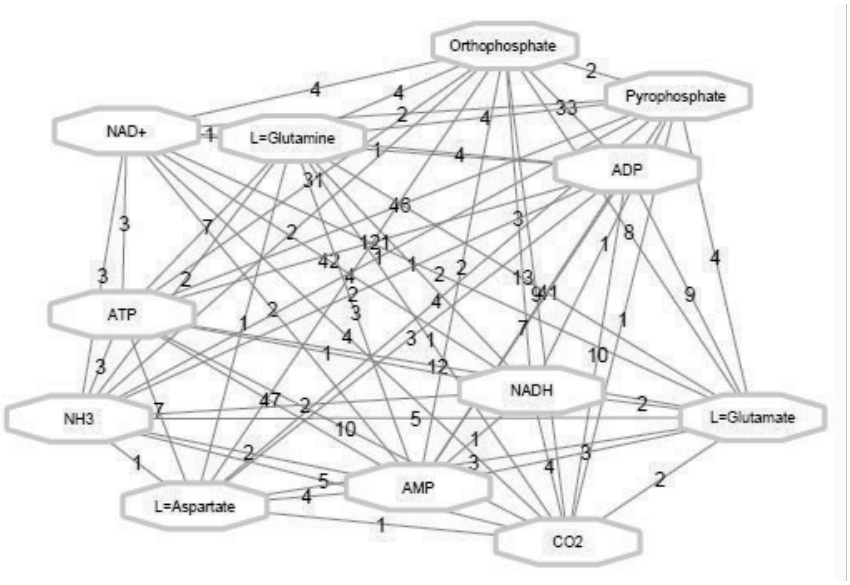
In order to confirm the reliability of the obtained results, particular information of the biochemical pathways of considered organism Saccha-

romyces cerevisiae are checked and confirmed with the data presented in Yeast Pathways Database.

Ammonia presented in the organism is used as a resource of nitrogen for amino-acid synthesis and if it released in larger quantity, it must be removed because of its toxicity. In the considered organism Saccharomyces cerevisiae, ammonia can be incorporated into the amino group of glutamate, by two pathways: the reductive amination of 2-ketoglutarate, catalyzed by glutamate dehydrogenase where NADPH serves as the source of electrons, or by the ATP-dependent synthesis of glutamine from glutamate and ammonia catalyzed by glutamine synthetase. Picture 7 visualize the results obtained by proposed method, for $k \in \{1,2\}$.

The first cluster in the left side of Picture 7 is a clique with 8 vertices and contains the main intermediates which figure in ammonia synthesis from glutamic and aspartic acids. On the right side of Picture 7 is shown the largest cluster obtained for $k = 2$. A wider set of intermediates is now shown, also including additional reactions.

More detailed graphical interpretation is shown in Picture 8, obtained by the proposed method for $k = 3$.



Picture 8 — Largest clusters obtained for k=3

Since the condition for forming clusters is now more relaxed, more intermediates figures in the cluster. In addition to the previous ones, in the cluster shown in Picture 8 we see the process of the oxidative deamination.

By this process, two toxic products are synthesized: hydrogen peroxide and ammonia. In Picture 8 we see that the method grouped all these intermediates in one cluster, which was not the case with the more strict conditions (cases $k = 1$ and $k = 2$).

Last, but not the least, data and results are the publicly available in the author's GitHub repository [12], in the supplemental data section.

5. TOPOLOGY SENSITIVE METAHEURISTIC OPTIMIZATION METHODS

The main motivation for integrating topology and metaheuristics comes from the notion that metaheuristics might use the topological regularities inside the solution space to better maneuver through it. This can become especially useful when the solution space becomes extremely large. In such situation, classical metaheuristics might use too much resources in order to search the solution space. Although this sounds like it could lead to premature convergence to local optima, we stress that our conceptual design essentially generalizes and encompasses the classical metaheuristic algorithms.

This means that the proposed metaheuristics, during its execution, gradually converge to its classical variants. Also, by imposing adequate parameters, these topologically sensitive metaheuristics can be used as a classical through its whole execution.

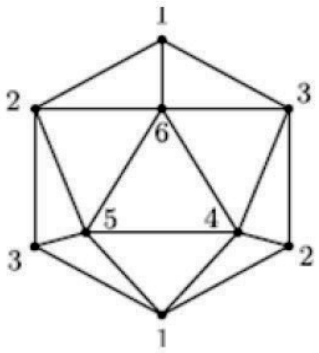
Execution of proposed topologically sensitive metaheuristics will resemble execution of any other metaheuristic: each execution creates a path in fitness landscape in order to reach global optima and avoids local optima. Therefore, fitness landscape analysis, which includes analysis of local optima positions, is very important for design of such metaheuristics. In other words, if some topological regularity in fitness landscape is detected, that regularity can be exploited and used for designing metaheuristic that will perform better than the alternatives. Topology-based models and techniques already achieved good results in revealing hidden structures and detecting new regularities [13], so it can be expected that it will be helpful in this domain.

In-depth discussion of the concepts described in this section are given in the paper [14].

The most important topology (more precisely, algebraic topology) concepts in this domain are simplicial complexes, homology groups and persistent homologies.

Simplicial complexes are combinatorial objects (abstract schemes) used from the early days of algebraic (combinatorial) topology as a bookkeeping device for triangulations of geometric objects. Conversely, they are used in the opposite direction for geometric presentation (visualization, geometric analysis and quantification) of the information (databases, point clouds) of any kind (not necessarily of geometric origin). The importance and versatility of simplicial complexes is illustrated by the fact that they appear under different names and in disguise in different areas of science, in and outside of mathematics. In cooperative game theory they are known as „simple games" (after John von Neumann and Oskar Morgenstern). A similar use is in social choice theory (reliability theory). We meet them as threshold complexes (of „short sets"), both in weighted voting games and in the geometry of configuration spaces of polygonal linkages (protein chains). Closely related concepts are monotone hypergraphs, monotone Boolean functions, finite partially ordered sets, etc.

For illustration, the hemi-icosahedron on Picture 9, triangulates the real projective plane and can be used for a combinatorial analysis of this object (homology calculation, non-embeddability in the 3-space, etc.). On the other hand, it provides an important example of a cooperative voting scheme (simple game) for six persons (parties), with 10 winning and 10 losing, 3-element coalitions, which is not realizable as a weighted voting scheme.



Picture 9 — hemi-icosahedron

Simplicial complexes provide historically the first foundation for the theory of homology groups, which capture the idea of higher (dis)connectivity (voids, holes) in geometric object. For example, the edge path 1–6–4–1 surrounds an essential 1-hole in the hemi-icosahedron, while if we traverse this edge-path twice, and perturb it to the edge-path 1–6-4–2-5–

1 (in the same homology class), we obtain a trivial 1-cycle (illustrating the torsion phenomenon in homology groups).

The so called persistent homology (initiated by A. Zamorodian, H. Edelsbrunner, G. Carlsson, and others) opened a new chapter of applications of homological methods and created a large part of what is today known as applied and computational algebraic topology. The importance of persistent homology is in its applicability to dynamical simplicial complexes (complexes depending on a parameter) which arise in the analysis of large databases (large finite metric spaces, point clouds, sparse matrices). Information collected from biological systems is typically less structured, and can be organized, via a concept of clustering into a dynamical (filtered) simplicial complex.

In order to show that topological enhancement can be done in general case (in top-down manner), two complementary techniques are selected: the first (VNS) is single-solution based and discrete coded metaheuristic, while the second (EM) is population-based and real coded metaheuristic. In paper [14], the conceptual design of the two topologically sensitive metaheuristics is presented: Topologically Sensitive Electromagnetism metaheuristics (TEM) and Topologically Sensitive Variable Neighborhood Search (TVNS).

5.1. TOPOLOGICALLY SENSITIVE ELECTROMAGNETISM-LIKE METAHEURISTICS

TEM is designed as a generalization of EM that builds on m-simplex data, where for special case $m = 0$ that algorithm becomes a classical VNS. The main difference in TEM, in comparison to classical EM, is in the movement step. For each solution point, within TEM, we try to find new solution position inside the solution space that will form a m-simplex with other m solution points from the current population (two variants: with or without that solution considered as a candidate). This should be done in such a way that average or maximal distance among 0-simplices is minimized.

More precisely, process starts with the most restrictive m-simplex, i. e. $m = m_{max}$. As mentioned in the description of the EM algorithm, the movement is controlled partially by forces that affect the solution point and partially by the randomness. In TEM, the movement is also controlled by forces, but now the randomness is restricted with respect to parameter m. This means that for $m > 0$, the set of possible positions from which the new position is randomly chosen now becomes smaller, i. e. it

becomes the subset of the set of possible positions where $m = 0$ (classical EM).

If, for a given solution point and current simplex size m, the movement is not possible, the simplex size is reduced by 1. Finally, if m reaches 0, that basically means that algorithm fell back to its classical version where every movement within distance threshold is allowed.

Note that this process of reducing the simplex size is done for each solution point separately. Also, note that the distance function threshold for deciding whether two 0-simplices are connected can be implicitly set by an algorithm to a sufficiently large value which will always allow a movement in the classical fallback EM.

The overall effect that we expect TEM will have on the search process in comparison to EM is increased preservation of the same or similar topological regularity through time (if this regularity exists). We also believe that the expansion of already existing simplices, especially large ones, is well motivated. This is because the existence of regular formation of local optima itself is an indicator that more new local optima may be found around that formation. Another important observation is that since TEM falls back to classical EM, we can expect that TEM will be generally applicable, i. e. if the topological regularity is low and cannot be exploited, TEM should work at least as good as classical EM (though performance might get deteriorated). Similar observation can be made for TVNS algorithm as well.

5.2. TOPOLOGICALLY SENSITIVE VARIABLE NEIGHBORHOOD SEARCH

TVNS is essentially conceived as a generalization of VNS that builds on m-simplex data (with special case $m = 0$ being a classical VNS). We will also sometimes refer to $(m + 1)$-simplex neighborhood which corresponds to collection of all valid simplices that can be formed by adding 0 -simplex to observed m-simplex. Therefore, 1-simplex neighborhood correspond to classical VNS, while m-simplex neighborhoods where $m > 0$ refer to its topological generalizations.

The main loop of TVNS should be made in such a way that the sequence of neighborhood structures, that are now parametrized by m and k, starts with the most restrictive neighborhood and after that proceeds with the sequence of more relaxed ones. Therefore, the neighborhoods will start with smallest neighborhood size $k = k_{min}$ and the most restrictive simplex structure $m = m_{max}$, and further proceed with reduction of m by 1. When

m reaches 0, it basically means that classical VNS algorithm is to be performed. After that, the k is increased by 1 and m is reset to m_{max}. The full cycle through neighborhoods is done when k reaches k_{max} and m reaches 0. If, at some moment, current solution is improved, both k and m are reset to its initial values.

6. CONCLUSION

Two metaheuristic optimization methods aimed at solving specific problems in bioinformatics and machine learning are described and the obtained results are analyzed. Topological enhancement for those methods are proposed.

Further research will be focused on theoretical characteristics of the proposed enchantments, on design and execution of computational experiments.

BIBLIOGRAPHY

[1] V. Filipović, A. Kartelj and D. Matić, „An electromagnetism metaheuristic for solving the Maximum Betweenness Problem," *Applied Soft Computing,* vol. 13, no. 2, pp. 1303–1313, 2013.

[2] M. Grbić, A. Kartelj, S. Janković, D. Matić and V. Filipović, „ Variable Neighborhood Search for Partitioning Sparse Biological Networks into the Maximum Edge-Weighted k-Plexes," *IEEE/ACM Transactions on Computational Biology and Bioinformatics,* vol. 17, no. 5, pp. 1822–1831, 2019.

[3] M. Grbić, D. Matić, A. Kartelj, S. Vračević and V. Filipović, „A three-phase method for identifying functionally related protein groups in weighted PPI networks," *Computational Biology and Chemistry,* vol. 86, 2020.

[4] P. Pardalos and E. Romeijn, Handbook of Optimization in Medicine, New York: Springer, 2009.

[5] V. Filipović, „Optimization, classification and dimensionality reduction in biomedicine and bioinformatics," *Biologia Serbica,* vol. 39, no. 1, pp. 83–98, 2017.

[6] A. Kartelj, „An Improved Electromagnetism-like Method for Feature Selection," *Journal of Multiple-Valued Logic and Soft Computing,* vol. 25, no. 2, pp. 169–187, 2015.

[7] I. Birbil and S. Fang, „An Electromagnetism-like Mechanism for Global Optimization," *Journal of Global Optimization,* vol. 25, pp. 263–282, 2003.

[8] C. Su and H. Lin, „Applying electromagnetism-like mechanism for feature selection," *Information Sciences,* vol. 181, no. 5, pp. 972–986, 2011.

[9] V. Filipovic, „An Electromagnetism Metaheuristic for the Uncapacitated Multiple Allocation Hub Location Problem," *Serdica Journal of Computing,* vol. 5, no. 3, pp. 261–272, 2011.

[10] A. Kartelj, N. Mitić, V. Filipović and D. Tošić, „Electromagnetism-like Algorithm for Support Vector Machine Parameter Tuning," *Soft Computing,* vol. 18, no. 10, pp. 1985–1998, 2013.
[11] N. Mladenović and P. Hansen, „Variable neighbourhood search," *Computers & Operational Research,* vol. 24, pp. 1097–1100, 1997.
[12] V. Filipović, „Suplemental info," 22 9 2022. [Online]. Available: https: // github. com/vladofilipovic/documents-science-public/tree/main/conferences/canu-2022/suplemental. [Accessed 22 9 2022].
[13] M. Reimann, M. Nolte, M. Scolamiero, K. Turner, R. Perin, G. Chindemi, P. Dotko, R. Levi, K. Hess and H. Makram, „Cliques of Neurons Bound into Cavities Provide a Missing Link between Structure and Function," *Frontiers in Computational Neuroscience,* vol. 11, 2017.
[14] A. Kartelj, V. Filipović, S. Vrećica and R. Živaljević, „Topologically sensitive metaheuristics," arhiv, 2020.