informatics

MDPI

Article

Discovering Entities Similarities in Biological Networks Using a Hybrid Immune Algorithm

Rocco A. Scollo, Antonio G. Spampinato, Georgia Fargetta, Vincenzo Cutello and Mario Pavone *

Department of Mathematics and Computer Science, University of Catania, Viale Andrea Doria 6, 95125 Catania, Italy

* Correspondence: mpavone@dmi.unict.it

Abstract: Disease phenotypes are generally caused by the failure of gene modules which often have similar biological roles. Through the study of biological networks, it is possible to identify the intrinsic structure of molecular interactions in order to identify the so-called *"disease modules"*. Community detection is an interesting and valuable approach to discovering the structure of the community in a complex network, revealing the internal organization of the nodes, and has become a leading research topic in the analysis of complex networks. This work investigates the link between biological modules and network communities in test-case biological networks that are commonly used as a reference point and which include *Protein–Protein Interaction Networks*, *Metabolic Networks* and *Transcriptional Regulation Networks*. In order to identify small and structurally well-defined communities in the biological context, a hybrid immune metaheuristic algorithm HYBRID-IA is proposed and compared with several metaheuristics, hyper-heuristics, and the well-known greedy algorithm LOUVAIN, with respect to modularity maximization. Considering the limitation of modularity optimization, which can fail to identify smaller communities, the reliability of HYBRID-IA was also analyzed with respect to three well-known sensitivity analysis measures (NMI, ARI and NVI) that assess how similar the detected communities are to real ones. By inspecting all outcomes and the performed comparisons, we will see that on one hand HYBRID-IA finds slightly lower modularity values than LOUVAIN, but outperforms all other metaheuristics, while on the other hand, it can detect communities more similar to the real ones when compared to those detected by LOUVAIN.

Keywords: metaheuristics; hybrid metaheuristics; community detection; networks biology; systems biology; immune algorithms; hybrid immune algorithms

Citation: Scollo, R.A.; Spampinato, A.G.; Fargetta, G.; Cutello, V.; Pavone, M. Discovering Entities Similarities in Biological Networks Using a Hybrid Immune Algorithm. *Informatics* **2023**, *10*, 18. https://doi.org/10.3390/informatics10010018

Academic Editor: Dmitry Zinoviev

Received: 31 October 2022
Revised: 31 December 2022
Accepted: 25 January 2023
Published: 31 January 2023

1. Introduction

In the last few years, significant advances in biological technology and the rapid adoption of high-throughput approaches [1] have made it possible the measurement of tens of thousands of "omic" data points across multiple levels (DNA, RNA, protein, metabolite, etc.) from a biological sample. The availability of a large amount of data has led to the development of new methods based on the integration analysis of molecular data with mathematical models, and the consequent consolidation of Systems Biology, an interdisciplinary research area that uses a holistic approach to biological research by using mathematical methods to derive global models of cellular processes in physiology and disease. The challenge from the systems biology point of view is to convert the data into information and knowledge so improve personalized therapies for patients. It becomes therefore necessary to carry out a high-level analysis of the representation of a biological system.

Networks are probably the most suitable tool for this type of investigation. They allow a good representation of the binary relationships between biological entities and are, for this reason, the most used tool in systems biology [2,3]. In particular, they also provide a mathematical representation of biological systems, where nodes may represent either

proteins, as in the case of protein–protein interaction networks, or genes and transcription factors, as in the case of gene co-expression networks, or metabolites, in the case of metabolic networks. These interactions determine the molecular and cellular mechanisms responsible for healthy or unhealthy states in organisms and can be physical or functional. In this second case, the network nodes are associated with each other in modules, where a tightly connected group of nodes share a common function to perform a specific task. The modules detected by biological networks are generally responsible for a common phenotype and are useful in providing insights related to biological functionality. Disease phenotypes are generally caused by the failure of groups of genes that are referred to as the disease form. Since the genes responsible for a phenotype often have common functions, there is a strong association between pathological and functional modules [3–5].

The detection of modules within biological networks, generally responsible for a common phenotype, is useful and crucial in providing insights into the biological function of these genes. The techniques that allow the identification of modules, known as *community detection* techniques, are methods that play a key role in obtaining the functional modules which appear to be closely related to pathological forms, whose recognition would be useful for the molecular understanding and the etiology of the disease. From such an understanding would arise the development of specific drugs whose targets would be the genes belonging to these modules.

In this research work, we propose a hybrid immune algorithm for community detection denoted HYBRID-IA. The algorithm is inspired by the immune system dynamics, and it carries out an effective exploration of large search spaces, by combining a random search process with a deterministic one, and by efficient exploitation of the gained knowledge. Although the algorithm has been successfully applied in several research areas, including community detection [6,7], the main goal of this work is to consolidate the robustness and reliability of the proposed algorithm in detecting community structures in large biological networks. Furthermore, to reach our goal, we studied the similarity of the detected communities to the real ones. Therefore, even though the presented algorithm is essentially the same, this research paper properly extends the work in [7] by carrying out a more detailed and complete sensitivity analysis in order to assess the real robustness of HYBRID-IA in detecting communities in complex and large networks, specifically larger and purely biological networks, which needed for the algorithm to be improved in terms of storage optimization and execution time.

The rest of the paper is organized as follows. Section 2 introduces and describes the community detection problem and its relevance in the biological context. Modularity optimization, which is the basis of the detection process, is described, and formally defined in Section 2. The proposed and developed HYBRID-IA algorithm is explained in detail in Section 3. In Section 4, we present a summary of the biological networks used as a data set, whilst outcomes and comparisons are displayed and discussed in Section 5. In particular, Section 5.1 shows the performed convergence and learning analysis, while Section 5.2 examines in detail all the obtained results and performed comparisons. In Section 5.3, we present the investigation we performed on HYBRID-IA with respect to the NMI evaluation metric. Furthermore, two other functional sensitivity metrics, namely ARI and NVI, were considered and tested, the results are presented in this section as well. Conclusions and final comments are presented in Section 6.

2. Community Detection

Community detection and clustering in networks are commonly used interchangeably. The goal of community detection is to find hidden relations among the nodes in the network and use such relations to identify clusters or communities of nodes. Generally speaking, we can define a community in a network or graph as a subset of the sets of nodes that identifies a highly dense connected subgraph with sparse connections with vertices outside the subgraph.

Community detection allows us to understand the dynamics of a complex network at different scales [8–10], such as connections and interactions between underlying entities, and, consequently, to uncover important information that becomes useful and crucial in many application areas such as biology; medicine; economic; social sciences; and many others. It is, however, a very difficult process due to the type of information modeled by the network. One important example is represented by social relationships among people. In such cases, the study of community structures in a network is a central issue in better understanding social dynamics, and, for this, it has inspired intense research activities also in closely related fields, such as patterns of people movements [11–15]. In biological sciences, the ability to detect highly linked communities and their interactions might be very beneficial in understanding how genes interact and affect each other.

Modularity (Q) is a valuation measure commonly used for assessing the quality of node partitions, i.e., communities, detected in a network [16]. Given such a valuation measure, the community detection problem can easily be summarized as the problem of finding clusters that maximize Q. The problem is computationally hard and, specifically, $\mathcal{NP}$-complete for its decisional version [17].

Several search algorithms for clustering problems have been developed and proven to be fast and reliable in finding as good communities as possible, even when dealing with large networks, solely based on specific topological properties of complex networks [18,19]. Some other algorithms make use of both topological and attribute information for the clustering analysis of complex networks. For example, in [20] the authors have proposed a fast fuzzy clustering algorithm, called F^{2}CAN, which integrates both types of information into a single fuzzy clustering framework [21] with a generalized momentum method to accelerate the convergence. Other algorithms, instead, are based on probabilistic models in which the problem of graph clustering is translated into a probabilistic inference problem, that is, a Bayesian model for attributed graph clustering [22,23].

In the context of biological networks, the authors in [24] propose a clustering framework that uses a variety of network motifs to identify functional modules based on the higher-order structure information included in a biological network, while in [25] they propose a fuzzy clustering algorithm which, after the semantic similarity between the attribute information of proteins is calculated, is used to identify protein complexes in a protein–protein interaction network.

Metaheuristics approaches have the ability to effectively tackle problems in complex, large and uncertain environments [26], thus they are more suitable for the optimization problems we are describing than exact methods. In this research paper we propose a *Hybrid Immunological Algorithm*, called HYBRID-IA, for the community detection problem, i.e., we apply a deterministic local search method within an evolutionary computation algorithm based on the workings of the immune system [27]. Briefly, the local search tries to improve the solutions found by the evolutionary mechanism. Using biological networks as a main case study, our goal is to prove the efficiency and robustness of HYBRID-IA in community detection.

Modularity Optimization in Networks

We do not expect a community structure in a graph to be randomly generated. Thus, communities in a graph can be detected by the difference of densities between the vertices of a given graph and the vertices of a random graph with the same size and same degree distribution.

We will use the following notations for the rest of the paper. Given an undirected graph $G = (V, E)$, with $N = |V|$ vertices and $M = |E|$ edges, its adjacency matrix A, and given a partition of the set of vertices V, that is the set of communities, $C = \{c_1, \ldots, c_k\}$,

- for each vertex $i \in V$, d_i will denote the degree of i, i.e., the total number of edges in E that have i as one of the endpoints;

- for each community c_h, d_{c_h} will denote the sum of the degrees of the vertices belonging to community c_h, i.e.,
$$d_{c_h} = \sum_{i \in c_h} d_i$$
- for each community c_h, ℓ_{c_h} will denote the number of edges or links inside the community c_h;
- for each vertex $i \in V$ and each community c_h, $\ell_{c_h}(i)$ will denote the number of edges or links from i to vertices in c_h.

Given a partition $C = \{c_1, \ldots, c_k\}$, let now $\delta(i,j)$ be a binary function that for every pair of vertices $i, j \in V$ has value $\delta(i,j) = 1$ if i and j belong to the same community and $\delta(i,j) = 0$ otherwise. The modularity value of such a partitioning C is defined as:

$$Q(C) = \frac{1}{2M}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\left(A_{ij} - \frac{d_i d_j}{2M}\right)\delta(i,j)\right], \tag{1}$$

where A_{ij} is the value at row i and column j of A.

An equivalent way to compute the modularity value is given by the following equation

$$Q(C) = \sum_{h=1}^{k}\left[\frac{\ell_{c_h}}{M} - \left(\frac{d_{c_h}}{2M}\right)^2\right], \tag{2}$$

We will use the latter equation when describing the workings of the local search operator. We note that in case of one single cluster, i.e., $|C| = 1$ and therefore $\delta(i,j) = 1$ for all pairs of nodes i and j, Equation (1) gives us the value 0, since $\sum_{i=1}^{N}\sum_{j=1}^{N} A_{ij} = \sum_{i=1}^{N} d_i = 2M$.

As proven in [17], the modularity value for any unweighted and undirected graphs G and any set of communities C verifies $-0.5 \leq Q(C) \leq 1$ and it represents a quality measure for a partition of a network into communities. A bad partitioning will result in a low value for Q implying the absence of real communities. High values of Q imply good partitions and, as a consequence, highly cohesive communities. As a consequence, community detection can be seen as the problem of maximizing Q, i.e., finding a clustering that will produce the highest value for Q as defined by Equation (1). As observed in [28], modularity optimization might not be able to identify communities that are small when compared with the network size and, in turn, produce large communities. Thus, the modules obtained through the process of modularity optimization must be checked and, possibly, rearranged.

3. HYBRID-IA: The Hybrid Immune Algorithm

Immune algorithms (IA) represent a set of computational systems, population-based metaheuristics inspired by the biological immune system and the way it acts to protect living organisms. They have been successfully applied in many search and optimization [29,30] and anomaly detection tasks [31]. The immune system has the ability to self-nonself discrimination and therefore detect, learn, and remember all the discovered foreign entities.

The hybrid immune algorithm HYBRID-IA, presented in [7], uses a local search procedure that deterministically tries to improve the found solutions. It is a well tested instance of a *Clonal Selection Algorithm* (CSA) [32,33], a particular class of immune algorithms, and so it is based on the two main concepts of antigen (Ag), which represents the problem to solve or optimize, and B cell, or antibody (Ab), which represents a candidate solution, i.e., a specific point in the solution space. For our specific community detection problem, given a graph $G = (V, E)$, a solution is a partition of the vertices, which will identify the clusters, i.e., communities. In particular, a B cell $\vec{x}$ is a sequence of $N = |V|$ integers, in the interval $[1, N]$, with the understanding that $x_i = h$ means that vertex i belongs to the cluster c_h.

At each time step t, the algorithm maintains a population $P^{(t)}$ of d candidate solutions. Every element of the population at time $t = 0$ is created by randomly assigning each vertex i to a group h, with $h \in [1, N]$. Equation (1) gives the fitness function of each element

$\vec{x} \in P^{(t)}$. The values is computed using the procedure ComputeFitness($P^{(t)}$). HYBRID-IA ends its evolution and terminates when it reaches the maximum number of generations (T_{max}). Algorithm 1 shows the pseudocode of HYBRID-IA.

Algorithm 1 Pseudo-code of HYBRID-IA.

```
1:  procedure HYBRID-IA(d, dup, ρ, τ_B, T_max)
2:      t ← 0
3:      P^(t) ← InitializePopulation(d)
4:      ComputeFitness(P^(t))
5:      while t ≤ T_max do
6:          P^(clo) ← Cloning(P^(t), dup)
7:          P^(hyp) ← Hypermutation(P^(clo), ρ)
8:          ComputeFitness(P^(hyp))
9:          (P_a^(t), P_a^(hyp)) ← Aging(P^(t), P^(hyp), τ_B)
10:         P^(select) ← (μ + λ)−Selection(P_a^(t), P_a^(hyp))
11:         P^(t+1) ← LocalSearch(P^(select))
12:         ComputeFitness(P^(t+1))
13:         t ← t + 1;
14:     end while
15: end procedure
```

The algorithm uses three immune operators, *cloning*, *hypermutation* and *aging*, which are described in what follows. The input parameters d (the number of elements in the population), dup (the number of duplications, i.e., clones, which are produced for each element), ρ (mutation rate), τ_B (maximum age allowed for each element), T_{max} (maximum number of iterations) are user defined.

Let us now describe in detail the workings of the operators.

3.1. The Cloning Operator

Cloning is the first immune operator to be called. Each B cell is copied dup times and so an intermediate population $P^{(clo)}$ of size $d \times dup$ is produced. We underline the fact that, to avoid premature convergence, the number of clones for each B cell is the same, i.e., it does not depend on its fitness value. To clarify, a number of close proportional to the fitness value of a B cell, could lead in a short amount of time to a population of B cells very similar to each other. This would in turn imply that the algorithm would not be able to perform a proper exploration of the search space, thus getting trapped in a local optimum.

Once a clone is created, HYBRID-IA assigns an age value to it. From such an assigned age until it reaches the maximum age allowed τ_B the clone would be able to remain in the population [34,35]. In more detail, a random age chosen in the range $[0 : \frac{2}{3}\tau_B]$ is assigned to each clone. As a consequence, each clone will stay in the population for at least $\frac{1}{3}\tau_B$ generations. The age assignment and the aging operator are able to keep the right amount of diversity among the solutions, helping the algorithm in avoiding premature convergences [36].

3.2. The Hypermutation Operator

The aim of the *hypermutation* operator is to efficiently and carefully explore the search space by acting on each clone in $P^{(clo)}$. In our case, if $\vec{x}$ is a cloned B cell, a mutation corresponds to the action of moving one node from a community to another.

The *mutation rate*, i.e., the number of mutations (changes) of each clone is inversely proportional to its fitness value. In other words, similarly to what happens with the natural immune system, the higher the fitness value of a clone, the smaller the number of mutations it will have. In more detail, if $\vec{x}$ is a cloned B cell, the mutation rate is the probability to move a node from one community to another and it is given by $\alpha = e^{-\rho \hat{f}(\vec{x})}$, where $\hat{f}(\vec{x})$ is

the fitness value of $\vec{x}$ normalized in the range $[0, 1]$ and ρ is a user-defined parameter that determines the shape of the mutation rate.

Figure 1 shows the different curves of the mutation rate behavior at the varying of the ρ parameter. One can see how the ρ parameter affects the probability α, at different fitness function values. It is possible to observe how low fitness values (which represent not good solutions) correspond to high α values, which means that a high number of nodes will be moved from one community to another; vice versa, at high fitness values (i.e., the best solutions), correspond low α values, and that is only a low number of nodes will be moved. Moreover, as expected, with lower values of ρ the values of α decrease more rapidly towards 0 when increasing normalized fitness values.

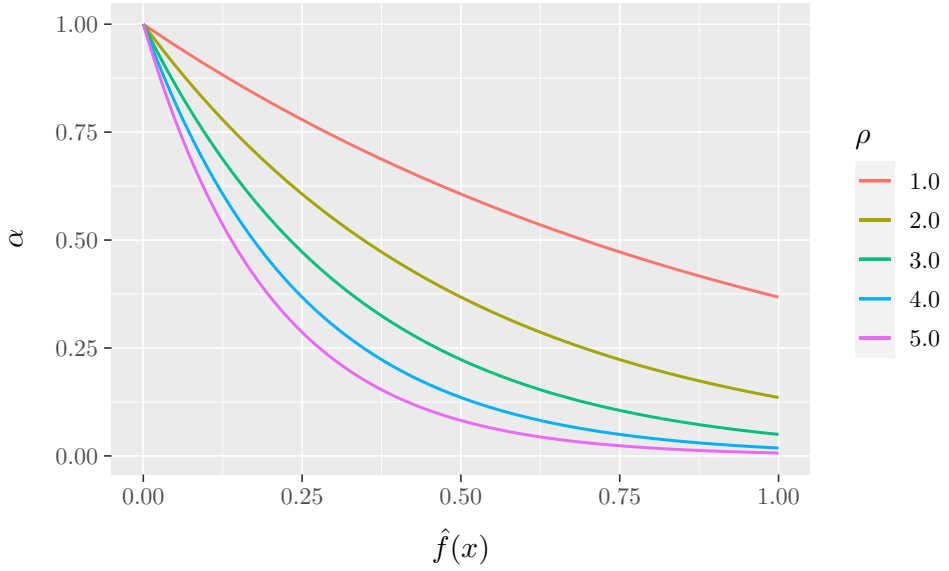


Figure 1. The α mutation curves at different values of ρ.

Formally, the hypermutation operator performs the following work. For each B cell, two different integers h and h' are randomly chosen in the range $[1, N]$. They represent two different communities, namely c_h and $c_{h'}$. The first one is chosen in particular among the existing community indices, whilst the latter is randomly chosen in the range $[1, N]$. Then, each of the vertices in c_h is moved into $c_{h'}$ with a probability given by α. In case $c_{h'}$ does not match any currently existing community, a new community will be created and added to the existing ones. Thus, depending on the value of h', i.e., whether or not it is the index of an existing community, two different mutations can occur: *merging* and *splitting*. In the first case, a subset of the vertices of community c_h will be merged with the vertices of the community $c_{h'}$. In the second case, instead, the community c_h will be split into two communities: c_h itself, and a new community $c_{h'}$. In Figure 2, we show for clarity a simple example of how these two cases work (merging in Figure 2a and splitting in Figure 2b).

The goal of the hypermutation operator is to discover new communities by selecting a variable percentage of nodes and moving them from their communities into others. The overall goal is to balance the effect of local search (described below) and, in turn, help the algorithm to avoid premature convergence.

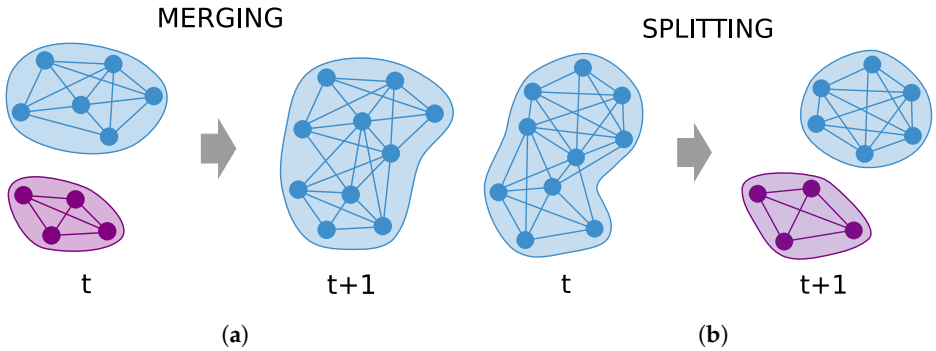


Figure 2. Hypermutation operator. (**a**) A subset of nodes from community c_h will be merged to an existing community $c_{h'}$. (**b**) A subset of nodes from community c_h will be split to create a new community $c_{h'}$.

3.3. The Static Aging Operator

The *static aging operator* is the one that plays a central role in the efficiency and robustness of HYBRID-IA, particularly when it is applied to complex and large problems. Its purpose is to assure a good turnover among the B cells and, as a consequence, to have high diversity in the population. It simply acts on each mutated B cell and decides, based on its age, whether to remove it or not from both $P^{(t)}$ and $P^{(hyp)}$, independently from its fitness value. Therefore, when the age of a B cell reaches the value $\tau_B + 1$, it will be removed from the population. The only exception, using the variant called *elitist aging operator*, could be made for the best current solution which could be kept in the population even if its age is higher than τ_B.

3.4. The Selection Operator

Among the B cells in both populations $P_a^{(t)}$ and $P_a^{(hyp)}$ that survived the aging phase, the $(\mu + \lambda)$-*Selection operator* with $\mu = d$ and $\lambda = (d \times dup)$, selects the best d elements. The elements of this population, $P^{(select)}$, will be the starting point to perform the local search. Since it selects the best d elements, the operator guarantees monotonicity of the population fitness evolution.

3.5. Local Search

Given a set of communities $C = \{c_1, \ldots, c_k\}$, if we move a vertex i from the community c_h to the community $c_{h'}$, we obtain a new set of communities C' which differs from C just on the placement of vertex i. The modularity of C' may differ from the modularity of C and such a variation, i.e., $\Delta Q_i(c_h, c_{h'}) = Q(C') - Q(C)$ is called *move gain*.

Using Equation (2) we have

$$\Delta Q_i(c_h, c_{h'}) = Q(C') - Q(C) = \frac{\ell_{c_{h'}}(i) - \ell_{c_h}(i)}{M} + d_i \left[\frac{d_{c_h} - d_i - d_{c_{h'}}}{2M^2} \right], \tag{3}$$

If $\Delta Q_i(c_h, c_{h'}) > 0$, then moving node i from the community c_h to the community $c_{h'}$ increases the modularity value. The *Move Vertex* (MV) (see [37]) methodology has the goal of finding a node i to move from its community c_h to an adjacent community $c_{h'}$ so to maximize ΔQ_i:

$$\underset{j \in Adj(i),\, j \in c_{h'}}{\operatorname{argmax}} \quad \Delta Q_i(c_h, c_{h'}), \tag{4}$$

where $i \in c_h$ and $Adj(i)$ is the adjacency list of node i.

The designed local search (LS) operator drives the algorithm towards more promising regions and, consequently, it properly speeds up its convergence. It explores the neighbour-

hood of each solution by using *Move Vertex*. In more detail, for each solution in $P^{(select)}$, we can compute for each community c_h of the solution the value

$$\chi(c_h) = \frac{\ell_{c_h}}{d_{c_h}},$$

i.e., the ratio between the total number of links inside the community and the sum of the degrees of its vertices.

Within each community c_h, we can compute for each of its vertices i the ratio

$$\chi(c_h, i) = \frac{\ell_{c_h}(i)}{d_i},$$

i.e., the ratio between the links with other vertices of the community and its degree. High values of $\chi(c_h)$ identify dense, well-formed communities while low values poorly formed communities. Similarly, high values of $\chi(c_h, i)$ identify those vertices which are strongly connected within their own community.

LS starts its work by sorting the communities in increasing order with respect to the ratio $\chi(c_h)$, and then, within each community c_h, it sorts the nodes in increasing order with respect to the values $\chi(c_h, i)$. Then, it applies MV on each community of the solution, starting from nodes that have at least an outgoing link, i.e., nodes that lie on the border of the community.

From a computational complexity point of view, we underline the fact that the values of Equation (3) can be efficiently computed because M and d_i are constants, using appropriate data structures the values of l_{c_h} and d_{c_h} can be quickly updated, while the terms $\ell_{c_h}(i)$ can be computed during when exploring all vertices adjacent to i. It follows that the moving vertex is linear on the dimension of the neighborhood of node i.

4. Biological Networks Data Set

In this section, we briefly describe the eight different biological networks we used during the tests and for which the communities were identified. They are grouped into the three types described below and refer to biological interactions and main molecular networks.

4.1. Protein–Protein Interaction (PPI) Networks

The physical interaction between proteins has always been an important consideration for gene functions. Proteins are the main participants in a variety of biological processes inside the cells, including signal transduction, homeostasis control, maintenance of internal balance and developmental processes [38]. They rarely function independently but form protein complexes [39]. The mathematical representation of the physical contacts between proteins inside the cell can be obtained through an undirected physical PPI network [40], in which nodes represent proteins and edges connect pairs of interacting proteins. By considering the spatial and temporal aspects of interactions, networks can help in understanding the general organization of protein–protein connections and in discovering the principles of their organization within the cell. PPI has a fundamental role in all biological processes and all organisms [1]. Therefore a complete knowledge of PPIs and their protein interconnections would allow the understanding of cell physiology both in pathogenic and normal states. In turn, this would have a great impact on disease diagnosis, since disease genes often interact with other disease genes [4], as well as for drug discovery and disease treatment [41–43]. In this work, two small *Cattle PPI* and *Helicobacter pylori PPI* Protein–Protein interactions [44–46] and two large networks (with a number of vertices greater than 2000), related to Yeast PPI instances [47,48] were considered. All networks in question are related to interactions between proteins in the three different organisms mentioned before (cattle, helicobacter pylori and yeast) where each node represents a protein and proteins are linked if they interact physically within the cell.

4.2. Metabolic Networks

Technological advancements and the possibility of sequencing whole genomes have made it possible not only to reconstruct the protein–protein interaction networks described above, but also to obtain the networks of biochemical reactions in many organisms. Metabolic networks are powerful tools for representing and studying a complete set of relationships between metabolites, small chemical compounds, and proteins/enzymes. They describe the set of processes and reactions that determine the biochemical and physiological properties of a cell, including the chemical reactions of metabolism, the metabolic pathways and regulatory interactions that drive these reactions. Metabolic networks make it possible to detect diseases given an enzymatic defect in a reaction that can affect flows in subsequent reactions. These defects often cause cascading effects responsible for associated metabolic diseases [49]. Therefore, this type of network can be used to understand if metabolic disorders are linked to their related reactions [50]. In order to investigate this functional information, it is necessary to identify the functional modules in it [51]. Identifying the communities in the metabolic networks will help in understanding the pathways and cycles in metabolic networks [52]. In the two real networks, we considered the metabolic networks of *Caenorhabditis elegans* and *E. coli bacteria* [53,54], each vertex represents a metabolite, and each direct link represents a reaction between them that binds the metabolite with the reaction product.

4.3. Transcriptional Regulatory Networks

Understanding the mechanisms underlying the regulation of gene expression is the main goal of contemporary biology. Important cellular processes, such as cell differentiation, cell cycle and metabolism are controlled by the complex biological mechanism of gene regulation. However, the relationship between structure and regulatory function is not easy to observe experimentally. Therefore, a System Biology-based approach is needed. In this regard, network theory is useful for understanding the activity behind these complex transcriptional regulatory mechanisms [55]. The transcriptional network can be represented as a directed graph, composed of transcription factors (TFs) and target genes (TGs) which are regulated in a tightly coordinated way.

Within the network, each node represents a gene and the edges represent direct transcriptional regulation. In case of prokaryotic organisms, each node represents an operon, i.e., a cluster of genes that are transcribed together to give a single messenger RNA (mRNA) molecule. Each edge is directed from a gene (or operon) that encodes a transcription factor to a gene (or operon) that is regulated by that transcription factor.

Transcription factors are modular proteins that regulate the gene expression of other proteins by binding to specific sites in the DNA (promoter sites) and allowing (or preventing) the synthesis of mRNA. The relationships between TFs and their targets (TGs) determine a given phenotype [56]. Moreover, in transcription regulatory networks, modules (or communities) correspond to sets of co-regulated genes [57–60]. For this reason, the problem of community detection plays a relevant role [61]. Escherichia coli and Saccharomyces cerevisiae are two well-known organisms often used as a model for studying gene regulation. In this work, we considered two transcriptional regulatory networks, *E. coli TRN* and *Yeast TRN* [62,63], they were built using transcription factors and target genes, and each edge in the network is directed from an operon that encodes a TF to an operon that it directly regulates.

4.4. Synthetic Networks

In addition to real biological networks, artificial instances were also taken into account in the experimental phase. Synthetic networks can be generated with different characteristics and with a known community structure. Using this kind of network, we can test algorithms on different scenarios and have the possibility to evaluate how good the detected communities are. The algorithm used to generate these synthetic networks is the *LFR benchmarks*, proposed in [64,65].

Mathematically, both the degree distributions and community size are power laws, with exponents τ_1 and τ_2, respectively. The mixing parameter μ_t, models and quantifies the relationship between the external and internal degree of each vertex. In more detail, each vertex of the networks shares a fraction $1 - \mu_t$ of its edges with the other vertices of its own community and the remaining fraction μ_t with vertices outside of its community. Furthermore, the LFR benchmarks can be used to generate directed and weighted synthetic networks with overlapping communities.

5. Experimental Results

In order to evaluate the efficiency and reliability of HYBRID-IA, we performed many experiments on all the biological networks above described. In each experiment HYBRID-IA maintains a population of $d = 100$ B cells; it uses a duplication parameter $dup = 2$; it keeps a solution for at most $\tau_B = 5$ generations within the population, and it uses $\rho = 1.0$ as mutation shape. Such a parameter setting is a consequence of some preliminary experimental results and of some specific knowledge previously acquired.

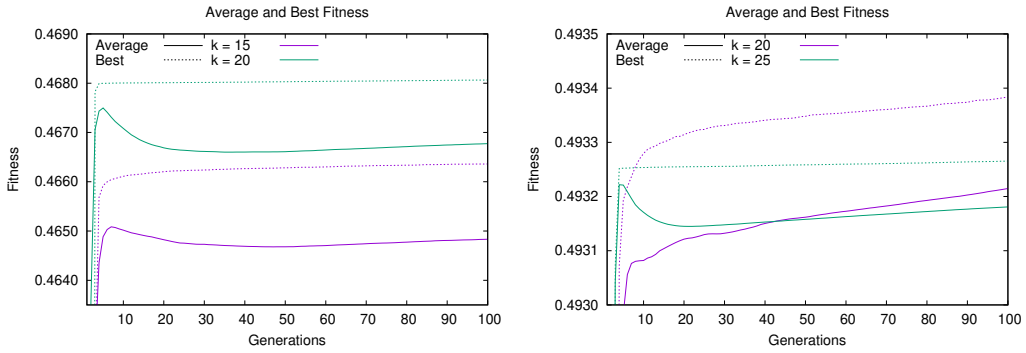
5.1. Convergence and Learning Analysis

In the initial part of the experimental phase, to better understand the efficiency of HYBRID-IA we focused our analysis on the convergence behavior and the learning rate. We chose artificial networks as benchmark instances. Such networks were generated using the LFR algorithm [64,65] and are described in Section 4.4.

In particular, networks with 1000 nodes and average degree 15 and 20, and networks with $|V| = 5000$ and average degree 20 and 25 were generated. For each of these generated networks, the maximum degree was set to 50, while the exponents of the power laws, which control the degree and community sizes distribution (τ_1 and τ_2), were set to 2 and 1, respectively. A minimum of 10 nodes to a maximum of 50 was set as sizes of the communities. The mixing parameter μ_t was fixed to 0.5. Finally, for these experiments, the maximum number of generations was set to $T_{max} = 100$ and 5 random instances were generated for each network parameters configuration.

Figure 3a shows the convergence plot on LFR instances with 1000 nodes and average degrees k of 15 (purple) and 20 (green). The two curves represent the best (dotted line) and average (continuous line) fitness of the population and both are averaged over 100 independent runs. From this plot we can observe how the two curves of the best fitness have the same trend for both values of k: they reach a high value of modularity in the early generations and then improve slowly. The improvement for $k = 20$ compared to the first generations is minimal, while for $k = 15$ the increase in modularity is slightly more significant. Instead, the average fitness curves have a similar trend in the first generations, but subsequently decrease and then gradually increase. From these two curves it can be seen how the population maintains a good degree of diversity, thus favoring a better exploration of the search space.

A similar situation can be also observed on the LFR instances with 5000 nodes. The plots in Figure 3b show the best and average fitness of the population for $k = 20$ (now purple) and $k = 25$ (green). For $k = 25$, HYBRID-IA obtains a high value of modularity within a few generations, and after that it stays in a steady state for the rest of the execution, reaching a high-modularity plateau [66]. On the other hand, for $k = 20$ the algorithm has a growth much more constant and linear, both in terms of the best solution and the average of the population. Furthermore, in this case, the two curves of best and average fitness are well separated, indicating that the algorithm maintains a good diversity of solutions within the population.



(**a**) (**b**)

Figure 3. Convergence behavior of HYBRID-IA: dotted lines are for best values, continuous lines for average values. Average and best fitness value versus generations on (**a**) `LFR(1000,15,0.5)` and `LFR(1000,20,0.5)`, and on (**b**) `LFR(5000,20,0.5)` and `LFR(5000,25,0.5)`.

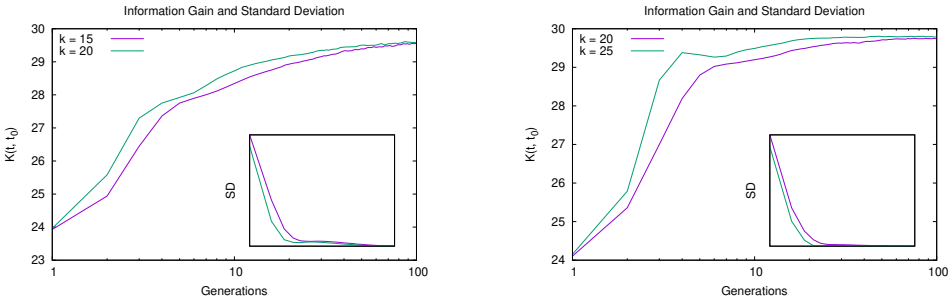
After the convergence behavior, we investigated the learning ability of HYBRID-IA. In order to do so, we used the quantity of information the algorithm gains during the evolutionary process [67,68], that is the learned amount of information compared to the randomly generated initial population. At each generation t, let B_m^t be the number of the B cells whose fitness function value is equal to m. The distribution function $f_m^{(t)}$ of the candidate solutions can therefore be defined as the ratio between B_m^t and the total number of candidate solutions:

$$f_m^{(t)} = \frac{B_m^t}{\sum_{m=0}^{h} B_m^t} = \frac{B_m^t}{d}. \tag{5}$$

It follows that the information gain $K(t, t_0)$ can be calculated as:

$$K(t, t_0) = \sum_m f_m^{(t)} \log(f_m^{(t)} / f_m^{(t_0)}). \tag{6}$$

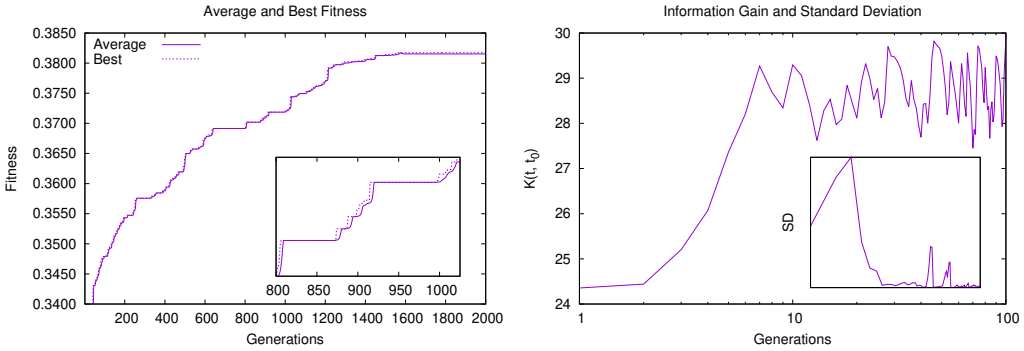
The plots in Figure 4 show the information gain obtained by the algorithm during its running in different scenarios. For both values of $|V|$, HYBRID-IA is able to learn information step by step, showing thus an increasing curve until it reaches a steady state, which is exactly when the modularity of all solutions begins to become similar. The monotonically increasing of the information gain curve until reaching a steady state is consistent with the *maximum information-gain principle*: $\frac{dK}{dt} \geq 0$. Overall, the convergence behavior and learning process analyzed (Figures 3 and 4), suggest that HYBRID-IA finds very quickly good solutions in networks with medium/high density (i.e., $k = 20$ for 1000 nodes and $k = 25$ for 5000), as the community structure is well-defined. On the other hand, on sparse networks, with an unclear community structure, the algorithm converges more slowly.



(**a**) (**b**)

Figure 4. Learning ability of HYBRID-IA: information gain and standard deviation versus generations on (**a**) `LFR(1000,15,0.5)` and `LFR(1000,20,0.5)`, and on (**b**) `LFR(5000,20,0.5)` and `LFR(5000,25,0.5)`.

In addition, we carried out a convergence analysis on the network *E. coli MRN* [54], which presents a very low density (see Table 1). Figure 5 shows the plots relative to the run in which HYBRID-IA has reached its best solution. Again, after the initial climb, the algorithm begins its exploration around the solutions found, improving gradually. In some places (inset plot of Figure 5a) the algorithm seems to stagnate in some local optima but, thanks to the *aging* operator, it manages to escape, and find better solutions. During these phases, the population tends to reduce its diversity, being almost entirely composed of solutions of equal quality.



(**a**) (**b**)

Figure 5. Convergence behavior of HYBRID-IA on *E. coli MRN* network. (**a**) Average and best fitness value of the population versus generations. (**b**) Information gain and standard deviation versus generations.

Table 1. The biological networks used in the experiments.

Name	Reference	$\|V\|$	$\|E\|$	$\Delta(G)$
Cattle PPI	[44]	268	303	0.85%
E. coli TRN	[62]	418	519	0.60%
C. elegans MRN	[53]	453	2025	1.98%
Yeast TRN	[63]	688	1078	0.46%
Helicobacter pylori PPI	[45,46]	724	1403	0.54%
E. coli MRN	[54]	1039	4741	0.88%
Yeast PPI (1)	[47]	2018	2705	0.13%
Yeast PPI (2)	[48]	2284	6646	0.25%

5.2. The Biological Networks

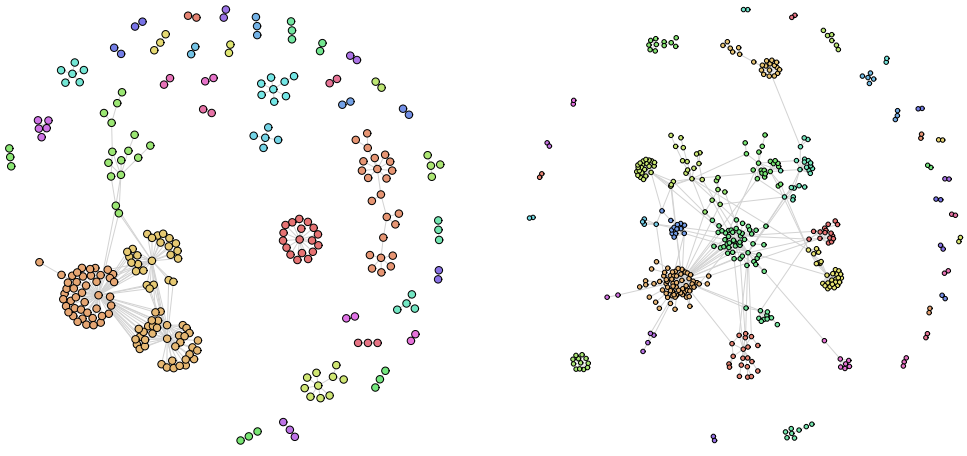
Metaheuristics and hyper-heuristics have been successfully applied to several complex network problems, and, specifically, in computational biology area [69–74]. Thus, we compare the results obtained by HYBRID-IA on the above-described biological networks with other well-known metaheuristics, all based on a modularity optimization approach [75]. In particular, we compare it with an effective Hyper-Heuristics Differential Search Algorithm (HDSA) [76] based on the random-walk movement used by an organism to migrate, in our case artificial superorganisms; an improved Bat Algorithm based on Differential Evolution algorithm (BADE) [77]; a Scatter Search algorithm based, an evolutionary method that has been successfully applied to hard optimization problems (SSGA) [78,79]; a modified Big Bang–Big Crunch algorithm (BB-BC) [80] an optimization method that relies on both the Big Bang and Big Crunch Theory and, consequently, energy dissipation increasing randomness and disorder, followed a random regrouping of particles; the original Bat Algorithm based on echolocation behavior of bats adapted for community detection (BA) [81]; a variation of the original Gravitational Search Algorithm, modified for solving the community detection problem (GSA) [82]; and the LOUVAIN algorithm [83], a greedy optimization method.

The characteristics of the biological networks are summarised in Table 1. In particular $\Delta(G)$ is the percentage of density of the corresponding graph, i.e., $\frac{2 \cdot |E|}{|V| \cdot (|V|-1)} \cdot 100$.

The parameter configuration used by HYBRID-IA is the same as described above, but the set number of generations (T_{max}) depends on the size of the biological network: for instance, for networks with less than 1000 nodes, T_{max} was set to 1000, while for networks with more than 1000 nodes, T_{max} was set to 2000. It is important to highlight that the results of all the other algorithms were taken from [75].

Table 2 displays the detailed results of HYBRID-IA compared with the other algorithms. It shows for each algorithm, the best values of the Q modularity (*Best*) found, the average of the values (*Mean*), the worst modularity (*Worst*), the standard deviation (*StD*) and the number of community structures (k) in the best solution. We stress the fact that unfortunately not all algorithms were tested on the same networks, as reported in the published results. Thus, some results are missing in the table. The results obtained by HYBRID-IA show its efficiency. Noticeably, the proposed HYBRID-IA algorithm outperforms all metaheuristics (i.e., excluding LOUVAIN) both in terms of the obtained values of modularity and mean, except for HDSA in the *E. coli TRN* biological network, although its best result is very close to that obtained by HDSA. Moreover, the average values obtained on *Cattle PPI*, *E. coli TRN*, *C. elegans MRN* and *Helicobacter pylori PPI* networks are better than the best modularity values obtained by the other algorithms, with the exception of the Hyper-heuristic Differential Search Algorithm (HDSA) and the best value obtained by BADE in *E. coli TRN*.

Let us now compare the results obtained by HYBRID-IA with the results obtained by LOUVAIN, the only greedy algorithm included in the comparison. It is clear how HYBRID-IA performs well equating the modularity value in the *Cattle PPI* dataset, and exceeding it in the *C. elegans MRN* and *E. coli MRN* networks. For these datasets, Figures 6and 7a,b show the detected community structures by HYBRID-IA. Figure 6b shows the communities generated for *E. coli TRN* network. For the other instances, although LOUVAIN performs generally better, the modularity reached by HYBRID-IA is however closer to the optimal one. Finally, as it will be explained in more detail in the next section, although LOUVAIN manages to achieve a better modularity value than HYBRID-IA, the latter reveals a higher number of communities. This is due to the different nature of the two algorithms since LOUVAIN tends to aggregate communities.

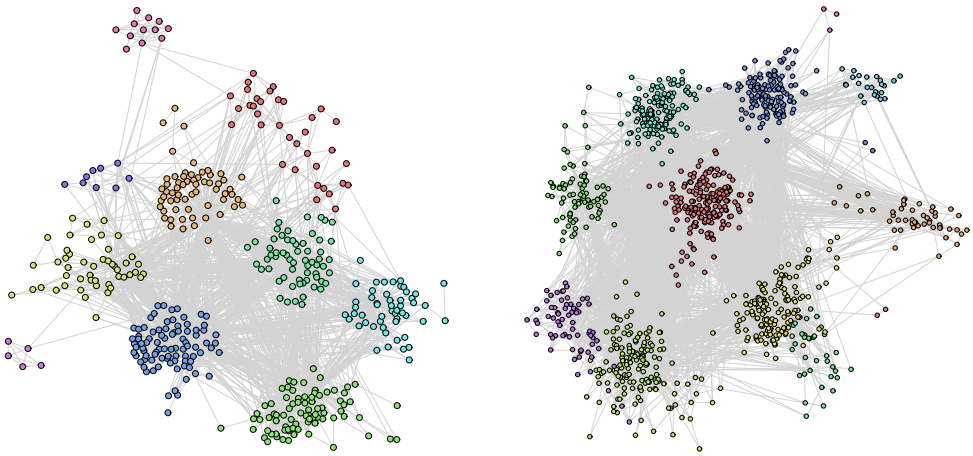


(**a**) (**b**)

Figure 6. Community structure identified by HYBRID-IA on (**a**) *Cattle PPI* and (**b**) *E. coli TRN* networks.

Table 2. Comparing the results obtained by HYBRID-IA on biological networks with the results obtained by other algorithms. The results are calculated over 100 independent runs for HYBRID-IA and LOUVAIN, while over 30 runs for the rest.

Name		HYBRID-IA	LOUVAIN	HDSA	BADE	SSGA	BB-BC	BA	GSA
Cattle PPI	*k*	40	40	40	41	40	48	42	43
	Best	0.7195	0.7195	0.7195	0.7183	0.7118	0.7095	0.7143	0.7053
	Worst	0.7011	0.7181	0.7194	0.7059	0.7052	0.7079	0.7063	0.6949
	Mean	0.7154	0.7193	0.7195	0.7138	0.7079	0.7084	0.7100	0.6983
	StD	0.0037	0.0005	0.0001	0.0051	0.0025	0.0007	0.0035	0.0041
E. coli TRN	*k*	43	41	47	58	61	71	56	61
	Best	0.7785	0.7793	0.7822	0.7680	0.7507	0.7520	0.7629	0.7416
	Worst	0.7563	0.7747	0.7808	0.7560	0.7412	0.7452	0.7542	0.7328
	Mean	0.7701	0.7779	0.7815	0.7621	0.7457	0.7485	0.7599	0.7375
	StD	0.0049	0.0011	0.0006	0.0043	0.0035	0.0026	0.0034	0.0034
C. elegans MRN	*k*	10	10	13	25	22	21	22	24
	Best	0.4506	0.4490	0.4185	0.3473	0.3336	0.3374	0.3514	0.3063
	Worst	0.4321	0.4216	0.3962	0.3335	0.3124	0.3194	0.3356	0.2974
	Mean	0.4437	0.4365	0.4074	0.3385	0.3220	0.3266	0.3438	0.3039
	StD	0.0040	0.0049	0.0010	0.0054	0.0077	0.0074	0.0073	0.0037
Yeast TRN	*k*	33	26	-	-	-	-	-	-
	Best	0.7668	0.7683	-	-	-	-	-	-
	Worst	0.7363	0.7489	-	-	-	-	-	-
	Mean	0.7569	0.7607	-	-	-	-	-	-
	StD	0.0050	0.0033	-	-	-	-	-	-
Helicobacter pylori PPI	*k*	51	24	52	69	70	75	62	77
	Best	0.5359	0.5462	0.5086	0.4926	0.4726	0.4681	0.4900	0.4600
	Worst	0.5104	0.5356	0.5048	0.4809	0.4659	0.4642	0.4738	0.4549
	Mean	0.5240	0.5410	0.5078	0.4854	0.4695	0.4660	0.4814	0.4567
	StD	0.0056	0.0025	0.0017	0.0047	0.0021	0.0018	0.0073	0.0020
E. coli MRN	*k*	13	8	-	-	-	-	-	-
	Best	0.3817	0.3734	-	-	-	-	-	-
	Worst	0.3598	0.3450	-	-	-	-	-	-
	Mean	0.3695	0.3583	-	-	-	-	-	-
	StD	0.0042	0.0058	-	-	-	-	-	-
Yeast PPI (2)	*k*	159	46	-	-	-	-	-	-
	Best	0.5796	0.5961	-	-	-	-	-	-
	Worst	0.5524	0.5870	-	-	-	-	-	-
	Mean	0.5652	0.5925	-	-	-	-	-	-
	StD	0.0052	0.0019	-	-	-	-	-	-
Yeast PPI (1)	*k*	353	213	-	-	-	-	-	-
	Best	0.7002	0.7648	-	-	-	-	-	-
	Worst	0.6602	0.7519	-	-	-	-	-	-
	Mean	0.6798	0.7609	-	-	-	-	-	-
	StD	0.0078	0.0022	-	-	-	-	-	-



(**a**) (**b**)

Figure 7. Community structure identified by HYBRID-IA on (**a**) *C. elegans MRN* and (**b**) *E. coli MRN* networks.

5.3. Functional Sensitivity of Community Detection

Modularity allows measuring how cohesive the detected communities are, but not to assess how similar they are to the real ones. Synthetic networks allow us to use another evaluation metric, namely the *Normalized Mutual Information* (NMI) [84], which is widely used to compare community detection methods since it is able to disclose the similarity between the genuine community (target) and the detected community structures. Since for the biological networks reported in Table 1 the real community structure is unknown, a new dataset of networks was used for this analysis. In particular, a new set of instances was generated using the LFR algorithm (see Section 4.4), with the mixing parameter μ_t that ranges from 0.1 to 0.8.

In Table 3, we can see the results of HYBRID-IA on these new synthetic datasets and we can compare them with the ones obtained by LOUVAIN. The features of the LFR networks tested are shown in the first column; for each of these lists of parameters, 5 random instances were generated. The values of modularity Q and NMI were computed over 100 independent runs for both algorithms.

Analyzing the comparison, it is possible to see how LOUVAIN outperforms HYBRID-IA in almost all networks with 1000 vertices with respect to the Q modularity metric, and in all instances with 5000 vertices. On the other hand, though, HYBRID-IA outperforms LOUVAIN with respect to the NMI index, in all networks except just for one $(1000, 20, 0.8)$.

What are the reasons behind such significant differences in the performances of the two algorithms? The combination of random search and local search that, along with the diversity in the population, introduced by the immune operators, require for HYBRID-IA a longer convergence time than LOUVAIN. LOUVAIN is a fast, multilevel algorithm that obtains good modularity values, but it aggregates communities too much, not reflecting the real community structures of the networks. These results prove, therefore, that HYBRID-IA is better able to detect communities that are closer to the true ones when compared with the greedy optimization algorithm. If on one hand modularity assesses the cohesion of the detected communities, on the other hand simply maximizing Q might not correspond to detecting true communities. Indeed, as it is also asserted in [28], when maximizing modularity it is possible to fail to identify smaller communities due to the degree of interconnection of the communities.

Table 3. Comparing HYBRID-IA and LOUVAIN on synthetic networks with 1000 and 5000 vertices, with respect to modularity (Q) and NMI evaluation metrics. In boldface are reported the best obtained values.

	HYBRID-IA		LOUVAIN			HYBRID-IA		LOUVAIN	
$(\|V\|, k, \mu_t)$	Q	*NMI*	Q	*NMI*	$(\|V\|, k, \mu_t)$	Q	*NMI*	Q	*NMI*
(1000, 15, 0.1)	0.8608	**0.9951**	0.8608	0.9918	(5000, 20, 0.1)	0.8923	**0.9988**	0.8934	0.9586
(1000, 15, 0.2)	0.7621	**0.9894**	0.7623	0.9807	(5000, 20, 0.2)	0.7927	**0.9965**	0.7949	0.9394
(1000, 15, 0.3)	0.6646	**0.9862**	0.6651	0.9716	(5000, 20, 0.3)	0.6929	**0.9965**	0.6960	0.9252
(1000, 15, 0.4)	0.5654	**0.9836**	0.5660	0.9691	(5000, 20, 0.4)	0.5931	**0.9948**	0.5976	0.9065
(1000, 15, 0.5)	0.4670	**0.9847**	0.4688	0.9462	(5000, 20, 0.5)	0.4936	**0.9951**	0.5003	0.8779
(1000, 15, 0.6)	0.3688	**0.9612**	0.3718	0.9113	(5000, 20, 0.6)	0.3939	**0.9966**	0.4030	0.8474
(1000, 15, 0.7)	0.2712	**0.5467**	0.2675	0.4977	(5000, 20, 0.7)	0.2932	**0.9927**	0.3056	0.8145
(1000, 15, 0.8)	0.2415	**0.1600**	0.2354	0.1536	(5000, 20, 0.8)	0.2084	**0.3285**	0.2102	0.2634
(1000, 20, 0.1)	0.8606	**0.9980**	0.8607	0.9931	(5000, 25, 0.1)	0.8922	**0.9993**	0.8925	0.9770
(1000, 20, 0.2)	0.7622	**0.9964**	0.7622	0.9914	(5000, 25, 0.2)	0.7925	**0.9988**	0.7936	0.9527
(1000, 20, 0.3)	0.6656	**0.9921**	0.6658	0.9830	(5000, 25, 0.3)	0.6929	**0.9986**	0.6948	0.9348
(1000, 20, 0.4)	0.5668	**0.9910**	0.5676	0.9656	(5000, 25, 0.4)	0.5931	**0.9987**	0.5966	0.9125
(1000, 20, 0.5)	0.4685	**0.9829**	0.4700	0.9491	(5000, 25, 0.5)	0.4934	**0.9955**	0.4983	0.8907
(1000, 20, 0.6)	0.3688	**0.9748**	0.3712	0.9263	(5000, 25, 0.6)	0.3939	**0.9950**	0.4008	0.8621
(1000, 20, 0.7)	0.2714	**0.9244**	0.2737	0.8230	(5000, 25, 0.7)	0.2940	**0.9951**	0.3037	0.8285
(1000, 20, 0.8)	0.2169	0.1708	0.2069	**0.1793**	(5000, 25, 0.8)	0.1872	**0.6067**	0.1942	0.5654

Figure 8 shows the curves of the NMI index for all LFR benchmarks with 1000 vertices. It can be observed that the two curves have the same trend: for low values of μ_t ($\leq$0.3), both algorithms obtain similar results, as the μ_t parameter increases, the gap between HYBRID-IA and LOUVAIN begins to be more consistent. However, when the mixing parameter assumes higher values, the generated LFR networks have community structures that are not well-defined, resulting, then, in low NMI values for both algorithms.

Figure 9, instead, shows the NMI curves for the LFR networks with 5000 nodes. For these instances, the difference between LOUVAIN and HYBRID-IA is much more substantial, and it is quite evident even at low values of μ_t.

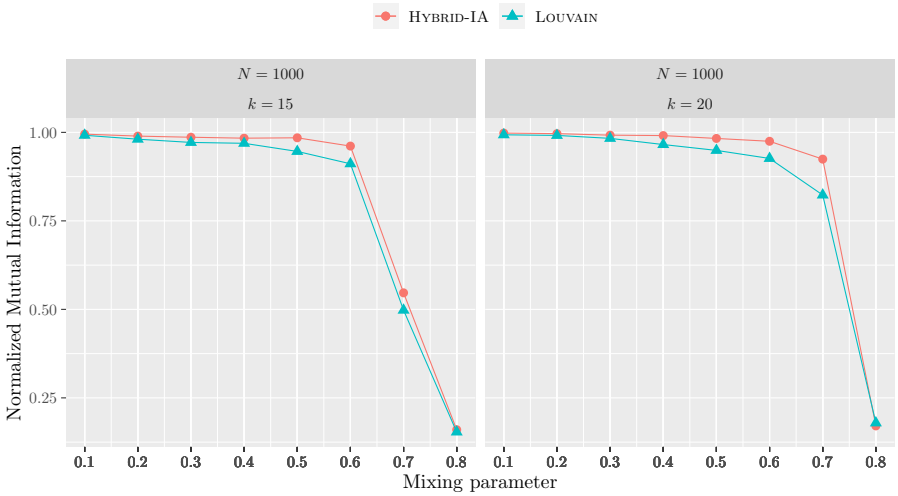


Figure 8. Performances of HYBRID-IA and LOUVAIN on the LFR instances with 1000 nodes and average degrees 15 and 20. The plots show the normalized mutual information as a function of the mixing parameter. Each point corresponds to an average over 5 produced graphs and 100 runs.

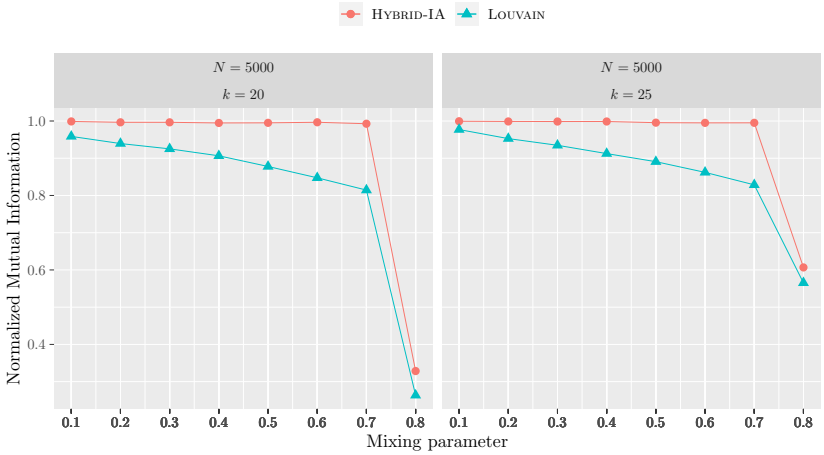


Figure 9. Performances of HYBRID-IA and LOUVAIN on the LFR instances with 5000 nodes and average degrees 20 and 25. The plots show the normalized mutual information as a function of the mixing parameter. Each point corresponds to an average over 5 produced graphs and 100 runs.

Following this modularity limitation, we conducted a more in-depth functional sensitivity analysis based on two other community structure similarity metrics: *Adjusted Rand Index* (ARI) [85], for measuring the similarity of partitions, and *Normalized Variation of Information* (NVI) [86], which is based on the Shannon entropy and measures the lost and the gained information in changing from one clustering to another one. It is worth recalling that while NMI and ARI values close to 1 denote strong similarity between the detected community and the real one (NMI $=$ ARI $=$ 1 identical communities), the NVI values, on the other hand, tend to 0 as the similarity between the compared communities increases (NVI $=$ 0 identical communities).

Thus, following the same experimental protocol, the three measures of functional sensitivity were computed on the synthetic networks considered above, with the addition of a larger one of $10,000$ vertices. The outcomes are reported in Table 4. By analyzing this table it is possible to assert that HYBRID-IA achieves better metric values than LOUVAIN in all three sensitivity measures. It follows therefore that, albeit LOUVAIN achieves slightly better modularity values in almost all networks considered (mainly on the larger ones), HYBRID-IA instead is able to detect communities strongly similar to the real ones outperforming LOUVAIN in all three sensitivity metrics. It is worth pointing out how HYBRID-IA obtains NMI and ARI values close to 1 in almost all tested networks, with particular reference to the largest one ($|V| = 10,000$).

Table 4. HYBRID-IA functional sensitivity analysis on synthetic networks at different sizes: from 1000 to $10,000$ vertices. NMI, ARI and NVI were considered as community structure similarity metrics. In boldface are reported the best obtained values.

	HYBRID-IA				**LOUVAIN**			
$(\|V\|, k, \mu_t)$	***Q***	***NMI***	***ARI***	***NVI***	***Q***	***NMI***	***ARI***	***NVI***
				$\|V\| = 1000$				
$(1000, 15, 0.1)$	0.8608	**0.9951**	**0.9873**	**0.0098**	0.8608	0.9918	0.9785	0.0161
$(1000, 15, 0.2)$	0.7621	**0.9894**	**0.9716**	**0.0209**	0.7623	0.9807	0.9483	0.0377
$(1000, 15, 0.3)$	0.6646	**0.9862**	**0.9567**	**0.0271**	0.6651	0.9716	0.9175	0.0550
$(1000, 15, 0.4)$	0.5654	**0.9836**	**0.9490**	**0.0322**	0.5660	0.9691	0.9104	0.0598
$(1000, 15, 0.5)$	0.4670	**0.9847**	**0.9467**	**0.0301**	0.4688	0.9462	0.8274	0.1019
$(1000, 15, 0.6)$	0.3688	**0.9612**	**0.8664**	**0.0742**	0.3718	0.9113	0.7368	0.1627
$(1000, 15, 0.7)$	0.2712	**0.5467**	**0.2379**	**0.6220**	0.2675	0.4977	0.2168	0.6664
$(1000, 15, 0.8)$	0.2415	**0.1600**	**0.0219**	**0.9130**	0.2354	0.1536	0.0218	0.9167

Table 4. *Cont.*

	HYBRID-IA				LOUVAIN			
$(\|V\|, k, \mu_t)$	Q	*NMI*	*ARI*	*NVI*	Q	*NMI*	*ARI*	*NVI*
(1000, 20, 0.1)	0.8606	**0.9980**	**0.9948**	**0.0040**	0.8607	0.9931	0.9842	0.0135
(1000, 20, 0.2)	0.7622	**0.9964**	**0.9918**	**0.0071**	0.7622	0.9914	0.9782	0.0170
(1000, 20, 0.3)	0.6656	**0.9921**	**0.9771**	**0.0156**	0.6658	0.9830	0.9525	0.0332
(1000, 20, 0.4)	0.5668	**0.9910**	**0.9707**	**0.0179**	0.5676	0.9656	0.8961	0.0664
(1000, 20, 0.5)	0.4685	**0.9829**	**0.9426**	**0.0337**	0.4700	0.9491	0.8363	0.0968
(1000, 20, 0.6)	0.3688	**0.9748**	**0.9038**	**0.0491**	0.3712	0.9263	0.7641	0.1370
(1000, 20, 0.7)	0.2714	**0.9244**	**0.7561**	**0.1399**	0.2737	0.8230	0.5403	0.3002
(1000, 20, 0.8)	0.2169	0.1708	0.0288	0.9064	0.2069	**0.1793**	**0.0300**	**0.9012**
$\|V\| = 5000$								
(5000, 20, 0.1)	0.8923	**0.9988**	**0.9940**	**0.0024**	0.8934	0.9586	0.8194	0.0794
(5000, 20, 0.2)	0.7927	**0.9965**	**0.9816**	**0.0070**	0.7949	0.9394	0.7302	0.1142
(5000, 20, 0.3)	0.6929	**0.9965**	**0.9808**	**0.0070**	0.6960	0.9252	0.6678	0.1392
(5000, 20, 0.4)	0.5931	**0.9948**	**0.9711**	**0.0104**	0.5976	0.9065	0.5941	0.1709
(5000, 20, 0.5)	0.4936	**0.9951**	**0.9728**	**0.0098**	0.5003	0.8779	0.4959	0.2177
(5000, 20, 0.6)	0.3939	**0.9966**	**0.9797**	**0.0068**	0.4030	0.8474	0.4048	0.2648
(5000, 20, 0.7)	0.2932	**0.9927**	**0.9539**	**0.0145**	0.3056	0.8145	0.3327	0.3130
(5000, 20, 0.8)	0.2084	**0.3285**	0.0176	**0.8027**	0.2102	0.2634	**0.0195**	0.8481
(5000, 25, 0.1)	0.8922	**0.9993**	**0.9966**	**0.0013**	0.8925	0.9770	0.8928	0.0449
(5000, 25, 0.2)	0.7925	**0.9988**	**0.9935**	**0.0024**	0.7936	0.9527	0.7821	0.0902
(5000, 25, 0.3)	0.6929	**0.9986**	**0.9921**	**0.0029**	0.6948	0.9348	0.7026	0.1225
(5000, 25, 0.4)	0.5931	**0.9987**	**0.9915**	**0.0027**	0.5966	0.9125	0.6158	0.1609
(5000, 25, 0.5)	0.4934	**0.9955**	**0.9747**	**0.0089**	0.4983	0.8907	0.5366	0.1970
(5000, 25, 0.6)	0.3939	**0.9950**	**0.9666**	**0.0100**	0.4008	0.8621	0.4473	0.2424
(5000, 25, 0.7)	0.2940	**0.9951**	**0.9653**	**0.0097**	0.3037	0.8285	0.3604	0.2927
(5000, 25, 0.8)	0.1872	**0.6067**	0.0667	**0.5607**	0.1942	0.5654	**0.1065**	0.6058
$\|V\| = 10,000$								
(10,000, 20, 0.1)	0.8938	**0.9995**	**0.9982**	**0.0010**	0.8945	0.9686	0.8874	0.0609
(10,000, 20, 0.2)	0.7938	**0.9981**	**0.9925**	**0.0037**	0.7951	0.9538	0.8198	0.0883
(10,000, 20, 0.3)	0.6940	**0.9980**	**0.9925**	**0.0040**	0.6960	0.9407	0.7563	0.1119
(10,000, 20, 0.4)	0.5941	**0.9977**	**0.9900**	**0.0045**	0.5972	0.9202	0.6682	0.1478
(10,000, 20, 0.5)	0.4942	**0.9988**	**0.9945**	**0.0024**	0.4986	0.8982	0.5793	0.1847
(10,000, 20, 0.6)	0.3943	**0.9996**	**0.9974**	**0.0008**	0.4004	0.8720	0.4872	0.2269
(10,000, 20, 0.7)	0.2909	**0.9847**	**0.8556**	**0.0301**	0.3013	0.8287	0.3737	0.2925
(10,000, 20, 0.8)	0.2064	**0.2621**	**0.0085**	**0.8491**	0.2094	0.1704	0.0079	0.9068
(10,000, 25, 0.1)	0.8937	**0.9995**	**0.9984**	**0.0010**	0.8940	0.9792	0.9253	0.0407
(10,000, 25, 0.2)	0.7940	**0.9993**	**0.9968**	**0.0014**	0.7947	0.9627	0.8524	0.0719
(10,000, 25, 0.3)	0.6941	**0.9990**	**0.9955**	**0.0020**	0.6955	0.9497	0.7946	0.0958
(10,000, 25, 0.4)	0.5942	**0.9992**	**0.9969**	**0.0015**	0.5964	0.9320	0.7179	0.1273
(10,000, 25, 0.5)	0.4944	**0.9982**	**0.9917**	**0.0036**	0.4978	0.9083	0.6199	0.1679
(10,000, 25, 0.6)	0.3943	**0.9984**	**0.9897**	**0.0032**	0.3989	0.8841	0.5313	0.2077
(10,000, 25, 0.7)	0.2941	**0.9981**	**0.9860**	**0.0038**	0.3008	0.8516	0.4299	0.2584
(10,000, 25, 0.8)	0.1849	**0.4316**	0.0227	**0.7245**	0.1867	0.3352	**0.0247**	0.7985

6. Conclusions

A *Hybrid Immune Algorithm*, called HYBRID-IA, was designed for the community detection problem and was tested on several large biological networks. The strength of HYBRID-IA is given not only by the immune operators (cloning, hypermutation and aging) but also by a specially designed Local Search, which aims to speed up the convergence of HYBRID-IA towards promising regions. Basically, it attempts deterministically to move a node from the belonging community to another within its neighbors with the purpose to refine and improve the solutions discovered.

In order to assess the robustness of HYBRID-IA, we performed several comparisons with other metaheuristics, hyper-heuristics and the well-known greedy algorithm LOUVAIN. We conducted a first comparison based on the *modularity function maximization*. However, due to the limitation of modularity optimization in detecting communities that are comparatively small, we performed other comparisons based on *Normalized Mutual Information* (NMI), *Adjusted Rand Index* (ARI) and *Normalized Variation of Information* (NVI), which are evaluation metrics commonly used in community detection that simply assess how similar the communities discovered are with respect to the real ones. All in all, after inspecting all the obtained outcomes and performing all the necessary comparisons, it clearly emerges how HYBRID-IA outperforms all metaheuristics and hyper-heuristics with which it was compared, in terms of best and mean modularity values. Focusing only on the comparison with LOUVAIN, it is possible to assert that although HYBRID-IA finds slightly lower modularity values, it is still able to detect communities more similar to the real ones than those discovered by LOUVAIN.

Some improvements, as future works, are currently in progress. We are addressing, in particular, the problem of improving the execution time and the searchability of HYBRID-IA especially on very large networks, through an adaptive multi-level approach. In addition, since the algorithm is based on several fixed and user-defined parameters, we believe that a further significant improvement of its searchability could be obtained by dynamically adapting such parameters through Machine Learning and/or Hyper-Heuristics techniques.

Author Contributions: All authors contributed equally to the methodology, investigation, validation and fomarl analysis, as well as to the writing—original draft preparation of the manuscript. V.C. and M.P. contributed to the writing, review and editing of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Acknowledgments: The authors would like to express their gratitude to the anonymous reviewers for their helpful feedback that measurably improved the manuscript. M. Pavone would like to thank the *Advanced New Technologies Research Laboratory* (ANTs lab.), Catania, Italy, for its kind support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Von Mering, C.; Krause, R.; Snel, B.; Cornell, M.; Oliver, S.G.; Fields, S.; Bork, P. Comparative assessment of large-scale data sets of protein–protein interactions. *Nature* **2002**, *417*, 399–403. [CrossRef] [PubMed]
2. Barabási, A.L.; Oltvai, Z.N. Network biology: Understanding the cell's functional organization. *Nat. Rev. Genet.* **2004**, *5*, 101–113. [CrossRef] [PubMed]
3. Barabási, A.L.; Gulbahce, N.; Loscalzo, J. Network medicine: A network-based approach to human disease. *Nat. Rev. Genet.* **2011**, *12*, 56–68. [CrossRef] [PubMed]
4. Goh, K.I.; Cusick, M.E.; Valle, D.; Childs, B.; Vidal, M.; Barabási, A.L. The human disease network. *Proc. Natl. Acad. Sci. USA* **2007**, *104*, 8685–8690. [CrossRef] [PubMed]
5. Aureli, M.; Masilamani, A.P.; Illuzzi, G.; Loberto, N.; Scandroglio, F.; Prinetti, A.; Chigorno, V.; Sonnino, S. Activity of plasma membrane β-galactosidase and β-glucosidase. *FEBS Lett.* **2009**, *583*, 2469–2473. [CrossRef]
6. Spampinato, A.G.; Scollo, R.A.; Cavallaro, S.; Pavone, M.; Cutello, V. An Immunological Algorithm for Graph Modularity Optimization. In *Advances in Intelligent Systems and Computing, Proceedings of the Advances in Computational Intelligence Systems (UKCI 2019), Portsmouth, UK, 4–6 September 2019*; Ju, Z., Yang, L., Yang, C.; Gegov, A., Zhou, D., Eds.; Springer: Cham, Switzerland, 2020; Volume 1043, pp. 235–247. [CrossRef]
7. Cutello, V.; Fargetta, G.; Pavone, M.; Scollo, R.A. Optimization Algorithms for Detection of Social Interactions. *Algorithms* **2020**, *13*, 139. [CrossRef]
8. Girvan, M.; Newman, M.E.J. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 7821–7826. [CrossRef]
9. Gulbahce, N.; Lehmann, S. The art of community detection. *BioEssays* **2008**, *30*, 934–938. [CrossRef]
10. Didimo, W.; Montecchiani, F. Fast layout computation of clustered networks: Algorithmic advances and experimental analysis. *Inf. Sci.* **2014**, *260*, 185–199. [CrossRef]

11. Buchin, K.; Buchin, M.; Gudmundsson, J.; Löffler, M.; Luo, J. Detecting commuting patterns by clustering subtrajectories. *Int. J. Comput. Geom. Appl.* **2011**, *21*, 253–282. [CrossRef]
12. Cavallaro, C.; Vizzari, G. A Novel Spatial–Temporal Analysis Approach to Pedestrian Groups Detection. *Procedia Comput. Sci.* **2022**, *207*, 2364–2373. [CrossRef]
13. Cavallaro, C.; Vitrià, J. Corridor Detection from Large GPS Trajectories Datasets. *Appl. Sci.* **2020**, *10*, 5003. [CrossRef]
14. Cavallaro, C.; Verga, G.; Tramontana, E.; Muscato, O. Multi-agent architecture for point of interest detection and recommendation. In Proceedings of the CEUR Workshop, Parma, Italy, 26–28 June 2019; Volume 2404, pp. 98–104.
15. Cavallaro, C.; Verga, G.; Tramontana, E.; Muscato, O. Eliciting cities points of interest from people movements and suggesting effective itineraries. *Intell. Artif.* **2020**, *14*, 49–61. [CrossRef]
16. Newman, M.E.J.; Girvan, M. Finding and evaluating community structure in networks. *Phys. Rev. E* **2004**, *69*, 026113. [CrossRef] [PubMed]
17. Brandes, U.; Delling, D.; Gaertler, M.; Görke, R.; Hoefer, M.; Nikoloski, Z.; Wagner, D. On Modularity Clustering. *IEEE Trans. Knowl. Data Eng.* **2008**, *20*, 172–188. [CrossRef]
18. Newman, M.E.J. Fast algorithm for detecting community structure in networks. *Phys. Rev. E* **2004**, *69*, 066133. [CrossRef]
19. Newman, M.E.J. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E* **2006**, *74*, 036104. [CrossRef] [PubMed]
20. Hu, L.; Pan, X.; Tang, Z.; Luo, X. A Fast Fuzzy Clustering Algorithm for Complex Networks via a Generalized Momentum Method. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 3473–3485. [CrossRef]
21. Hu, L.; Chan, K.C.C. Fuzzy Clustering in a Complex Network Based on Content Relevance and Link Structures. *IEEE Trans. Fuzzy Syst.* **2016**, *24*, 456–470. [CrossRef]
22. Xu, Z.; Ke, Y.; Wang, Y.; Cheng, H.; Cheng, J. GBAGC: A General Bayesian Framework for Attributed Graph Clustering. *ACM Trans. Knowl. Discov. Data* **2014**, *9*, 1–43. [CrossRef]
23. Hu, L.; Zhang, J.; Pan, X.; Yan, H.; You, Z.H. HiSCF: Leveraging higher-order structures for clustering analysis in biological networks. *Bioinformatics* **2020**, *37*, 542–550. [CrossRef]
24. Hu, L.; Chan, K.C.C.; Yuan, X.; Xiong, S. A Variational Bayesian Framework for Cluster Analysis in a Complex Network. *IEEE Trans. Knowl. Data Eng.* **2020**, *32*, 2115–2128. [CrossRef]
25. Pan, X.; Hu, L.; Hu, P.; You, Z.H. Identifying Protein Complexes From Protein–Protein Interaction Networks Based on Fuzzy Clustering and GO Semantic Information. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2022**, *19*, 2882–2893. [CrossRef] [PubMed]
26. Talbi, E.G. *Metaheuristics: From Design to Implementation*; Wiley Publishing: Hoboken, NJ, USA, 2009.
27. Scollo, R.A.; Cutello, V.; Pavone, M. Where the Local Search Affects Best in an Immune Algorithm. In *Lecture Notes in Artificial Intelligence, Proceedings of the AIxIA 2020—Advances in Artificial Intelligence (AIxIA 2020), Virtual, 25–27 November 2020*; Baldoni, M., Bandini, S., Eds.; Springer: Cham, Swizterland, 2021; Volume 12414, pp. 99–114. [CrossRef]
28. Fortunato, S.; Barthélemy, M. Resolution limit in community detection. *Proc. Natl. Acad. Sci. USA* **2007**, *104*, 36–41. [CrossRef] [PubMed]
29. Costanza, J.; Cutello, V.; Pavone, M. A Memetic Immunological Algorithm for Resource Allocation Problem. In *Lecture Notes in Computer Science, Proceedings of the 10th International Conference on Artificial Immune Systems (ICARIS 2011), Cambridge, UK, 18–21 July 2011*; Liò, P., Nicosia, G., Stibor, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6825, pp. 308–320. [CrossRef]
30. Stracquadanio, G.; Greco, O.; Conca, P.; Cutello, V.; Pavone, M.; Nicosia, G. Packing equal disks in a unit square: An immunological optimization approach. In Proceedings of the International Workshop on Artificial Immune Systems (AIS), Taormina-Sicily, Italy, 17–18 July 2015; pp. 1–5. [CrossRef]
31. Fouladvand, S.; Osareh, A.; Shadgar, B.; Pavone, M.; Sharafi, S. DENSA: An effective negative selection algorithm with flexible boundaries for self-space and dynamic number of detectors. *Eng. Appl. Artif. Intell.* **2017**, *62*, 359–372. [CrossRef]
32. Pavone, M.; Narzisi, G.; Nicosia, G. Clonal selection: An immunological algorithm for global optimization over continuous spaces. *J. Glob. Optim.* **2012**, *53*, 769–808. [CrossRef]
33. Cutello, V.; Oliva, M.; Pavone, M.; Scollo, R.A. An Immune Metaheuristics for Large Instances of the Weighted Feedback Vertex Set Problem. In Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI 2019), Xiamen, China, 6–9 December 2019; pp. 1928–1936. [CrossRef]
34. Di Stefano, A.; Vitale, A.; Cutello, V.; Pavone, M. How long should offspring lifespan be in order to obtain a proper exploration? In Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI 2016), Athens, Greece, 6–9 December 2016; pp. 1–8. [CrossRef]
35. Vitale, A.; Di Stefano, A.; Cutello, V.; Pavone, M. The Influence of Age Assignments on the Performance of Immune Algorithms. In *Advances in Computational Intelligence Systems, Proceedings of the 18th UK Workshop on Computational Intelligence, Nottingham, UK, 5–7 September 2018*; Lotfi, A., Bouchachia, H.; Gegov, A., Langensiepen, C., McGinnity, M., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 16–28.
36. Cutello, V.; Nicosia, G.; Pavone, M.; Stracquadanio, G. An Information-Theoretic Approach for Clonal Selection Algorithms. In *Lecture Notes in Computer Science, Proceedings of the 9th International Conference on Artificial Immune Systems (ICARIS 2010), Edinburgh, UK, 26–29 July 2010*; Hart, E., McEwan, C.; Timmis, J., Hone, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6209, pp. 144–157. [CrossRef]
37. Kernighan, B.W.; Lin, S. An efficient heuristic procedure for partitioning graphs. *Bell Syst. Tech. J.* **1970**, *49*, 291–307. [CrossRef]

38. Zhang, Y.; Gao, P.; Yuan, J.S. Plant Protein–Protein Interaction Network and Interactome. *Curr. Genom.* **2010**, *11*, 40–46. [CrossRef]
39. Gu, H.; Zhu, P.; Jiao, Y.; Meng, Y.; Chen, M. PRIN: A predicted rice interactome network. *BMC Bioinform.* **2011**, *12*, 161. [CrossRef]
40. Lim, Y.H.; Charette, J.M.; Baserga, S.J. Assembling a Protein–Protein Interaction Map of the SSU Processome from Existing Datasets. *PLoS ONE* **2011**, *6*, e17701. [CrossRef]
41. Mullard, A. Protein–protein interaction inhibitors get into the groove. *Nat. Rev. Drug Discov.* **2012**, *11*, 173–175. [CrossRef] [PubMed]
42. Taylor, I.W.; Linding, R.; Warde-Farley, D.; Liu, Y.; Pesquita, C.; Faria, D.; Bull, S.; Pawson, T.; Morris, Q.; Wrana, J.L. Dynamic modularity in protein interaction networks predicts breast cancer outcome. *Nat. Biotechnol.* **2009**, *27*, 199–204. [CrossRef] [PubMed]
43. Diss, G.; Filteau, M.; Freschi, L.; Leducq, J.B.; Rochette, S.; Torres-Quiroz, F.; Landry, C.R. Integrative avenues for exploring the dynamics and evolution of protein interaction networks. *Curr. Opin. Biotechnol.* **2013**, *24*, 775–783. [CrossRef] [PubMed]
44. Cattle Protein–Protein Interactions, 2009. Available online: https://biit.cs.ut.ee/graphweb/welcome.cgi?t=examples (accessed on 30 October 2022).
45. Xenarios, I.; Rice, D.W.; Salwinski, L.; Baron, M.K.; Marcotte, E.M.; Eisenberg, D. DIP: The Database of Interacting Proteins. *Nucleic Acids Res.* **2000**, *28*, 289–291. [CrossRef]
46. Rain, J.C.; Selig, L.; De Reuse, H.; Battaglia, V.; Reverdy, C.; Simon, S.; Lenzen, G.; Petel, F.; Wojcik, J.; Schächter, V.; et al. The protein—Protein interaction map of Helicobacter pylori. *Nature* **2001**, *409*, 211–215. [CrossRef] [PubMed]
47. Yu, H.; Braun, P.; Yildirim, M.A.; Lemmens, I.; Venkatesan, K.; Sahalie, J.; Hirozane-Kishikawa, T.; Gebreab, F.; Li, N.; Simonis, N.; et al. High-Quality Binary Protein Interaction Map of the Yeast Interactome Network. *Science* **2008**, *322*, 104–110. [CrossRef] [PubMed]
48. Bu, D.; Zhao, Y.; Cai, L.; Xue, H.; Zhu, X.; Lu, H.; Zhang, J.; Sun, S.; Ling, L.; Zhang, N.; et al. Topological structure analysis of the protein–protein interaction network in budding yeast. *Nucleic Acids Res.* **2003**, *31*, 2443–2450. [CrossRef]
49. Lee, D.S.; Park, J.; Kay, K.A.; Christakis, N.A.; Oltvai, Z.N.; Barabási, A.L. The implications of human metabolic network topology for disease comorbidity. *Proc. Natl. Acad. Sci. USA* **2008**, *105*, 9880–9885. [CrossRef]
50. Ross, R.; Dagnone, D.; Jones, P.J.H.; Smith, H.; Paddags, A.; Hudson, R.; Janssen, I. Reduction in Obesity and Related Comorbid Conditions after Diet-Induced Weight Loss or Exercise-Induced Weight Loss in Men. *Ann. Intern. Med.* **2000**, *133*, 92–103. [CrossRef]
51. Yanrui, D.; Zhen, Z.; Wenchao, W.; Yujie, C. Identifying the Communities in the Metabolic Network Using 'Component' Definition and Girvan-Newman Algorithm. In Proceedings of the 14th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES 2015), Guiyang, China, 18–24 August 2015; pp. 42–45. [CrossRef]
52. Fortunato, S. Community detection in graphs. *Phys. Rep.* **2010**, *486*, 75–174. [CrossRef]
53. Duch, J.; Arenas, A. Community detection in complex networks using extremal optimization. *Phys. Rev. E* **2005**, *72*, 027104. [CrossRef] [PubMed]
54. Schellenberger, J.; Park, J.O.; Conrad, T.M.; Palsson, B.Ø. BiGG: A Biochemical Genetic and Genomic knowledgebase of large scale metabolic reconstructions. *BMC Bioinform.* **2010**, *11*, 213. [CrossRef] [PubMed]
55. Barabási, A.L.; Albert, R.; Jeong, H. Scale-free characteristics of random networks: The topology of the world-wide web. *Phys. A Stat. Mech. Appl.* **2000**, *281*, 69–77. [CrossRef]
56. Ravasi, T.; Suzuki, H.; Cannistraci, C.V.; Katayama, S.; Bajic, V.B.; Tan, K.; Akalin, A.; Schmeier, S.; Kanamori-Katayama, M.; Bertin, N.; et al. An Atlas of Combinatorial Transcriptional Regulation in Mouse and Man. *Cell* **2010**, *140*, 744–752. [CrossRef]
57. Cantini, L.; Medico, E.; Fortunato, S.; Caselle, M. Detection of gene communities in multi-networks reveals cancer drivers. *Sci. Rep.* **2015**, *5*, 17386. [CrossRef] [PubMed]
58. Marbach, D.; Costello, J.C.; Küffner, R.; Vega, N.M.; Prill, R.J.; Camacho, D.M.; Allison, K.R.; Kellis, M.; Collins, J.J.; Stolovitzky, G. Wisdom of crowds for robust gene network inference. *Nat. Methods* **2012**, *9*, 796–804. [CrossRef] [PubMed]
59. Wilkinson, D.M.; Huberman, B.A. A method for finding communities of related genes. *Proc. Natl. Acad. Sci. USA* **2004**, *101*, 5241–5248. [CrossRef]
60. Zhu, J.; Zhang, B.; Smith, E.N.; Drees, B.; Brem, R.B.; Kruglyak, L.; Bumgarner, R.E.; Schadt, E.E. Integrating large-scale functional genomic data to dissect the complexity of yeast regulatory networks. *Nat. Genet.* **2008**, *40*, 854–861. [CrossRef]
61. Tang, B.; Hsu, H.K.; Hsu, P.Y.; Bonneville, R.; Chen, S.S.; Huang, T.H.M.; Jin, V.X. Hierarchical Modularity in ERα Transcriptional Network Is Associated with Distinct Functions and Implicates Clinical Outcomes. *Sci. Rep.* **2012**, *2*, 875. [CrossRef]
62. Shen-Orr, S.S.; Milo, R.; Mangan, S.; Alon, U. Network motifs in the transcriptional regulation network of Escherichia coli. *Nat. Genet.* **2002**, *31*, 64–68. [CrossRef]
63. Milo, R.; Shen-Orr, S.; Itzkovitz, S.; Kashtan, N.; Chklovskii, D.; Alon, U. Network Motifs: Simple Building Blocks of Complex Networks. *Science* **2002**, *298*, 824–827. [CrossRef] [PubMed]
64. Lancichinetti, A.; Fortunato, S.; Radicchi, F. Benchmark graphs for testing community detection algorithms. *Phys. Rev. E* **2008**, *78*, 046110. [CrossRef] [PubMed]
65. Lancichinetti, A.; Fortunato, S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E* **2009**, *80*, 016118. [CrossRef] [PubMed]
66. Good, B.H.; De Montjoye, Y.A.; Clauset, A. Performance of modularity maximization in practical contexts. *Phys. Rev. E* **2010**, *81*, 046106. [CrossRef]

67. Kullback, S. *Information Theory and Statistics*; Wiley: Hoboken, NJ, USA, 1959.
68. Cutello, V.; Nicosia, G.; Pavone, M.; Stracquadanio, G. Entropic divergence for population based optimization algorithms. In Proceedings of the IEEE Congress on Evolutionary Computation, Barcelona, Spain, 18–23 July 2010; pp. 1–8. [CrossRef]
69. Zito, F.; Cutello, V.; Pavone, M. Optimizing Multi-Variable Time Series Forecasting using Metaheuristics. In *Lecture Notes in Computer Science, Proceedings of the 14th Metaheuristics International Conference (MIC 2022), Ortigia-Syracuse, Italy, 11–14 July 2022*; Di Gaspero, L., Festa, P., Nakib, A., Pavone, M., Eds.; Springer: Cham, Swizterland, 2023; Volume 13838.
70. Altuntas, V.; Gök, M.; Kahveci, T. Stability Analysis of Biological Networks' Diffusion State. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2020**, *17*, 1406–1418. [CrossRef]
71. Penas, D.R.; González, P.; Egea, J.A.; Doallo, R.; Banga, J.R. Parameter estimation in large-scale systems biology models: A parallel and self-adaptive cooperative strategy. *BMC Bioinform.* **2017**, *18*, 52. [CrossRef]
72. Barros, R.C.; Winck, A.T.; Machado, K.S.; Basgalupp, M.P.; de Carvalho, A.C.; Ruiz, D.D.; de Souza, O.N. Automatic design of decision-tree induction algorithms tailored to flexible-receptor docking data. *BMC Bioinform.* **2012**, *13*, 310. [CrossRef]
73. Cutillas-Lozano, J.M.; Giménez, D.; Almeida, F. Hyperheuristics Based on Parametrized Metaheuristic Schemes. In Proceedings of the Annual Conference on Genetic and Evolutionary Computation (GECCO '15), Madrid Spain, 11–15 July 2015; Association for Computing Machinery: New York, NY, USA, 2015; pp. 361–368. [CrossRef]
74. Bonidia, R.P.; Avila Santos, A.P.; de Almeida, B.L.S.; Stadler, P.F.; da Rocha, U.N.; Sanches, D.S.; de Carvalho, A.C.P.L.F. BioAutoML: Automated feature engineering and metalearning to predict noncoding RNAs in bacteria. *Briefings Bioinform.* **2022**, *23*, 1–13. [CrossRef]
75. Atay, Y.; Koc, I.; Babaoglu, I.; Kodaz, H. Community detection from biological and social networks: A comparative analysis of metaheuristic algorithms. *Appl. Soft Comput.* **2017**, *50*, 194–211. [CrossRef]
76. Civicioglu, P. Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. *Comput. Geosci.* **2012**, *46*, 229–247. [CrossRef]
77. Storn, R.; Price, K. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
78. Glover, F. Heuristics for integer programming using surrogate constraints. *Decis. Sci.* **1977**, *8*, 156–166. [CrossRef]
79. Martí, R.; Laguna, M.; Glover, F. Principles of scatter search. *Eur. J. Oper. Res.* **2006**, *169*, 359–372. [CrossRef]
80. Erol, O.K.; Eksin, I. A new optimization method: Big Bang–Big Crunch. *Adv. Eng. Softw.* **2006**, *37*, 106–111. [CrossRef]
81. Yang, X.S. A New Metaheuristic Bat-Inspired Algorithm. In *Studies in Computational Intelligence, Proceedings of the Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), Granada, Spain, 12–14 May 2010*; González, J.R., Pelta, D.A., Cruz, C., Terrazas, G., Krasnogor, N., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74. [CrossRef]
82. Rashedi, E.; Nezamabadi-pour, H.; Saryazdi, S. GSA: A Gravitational Search Algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
83. Blondel, V.D.; Guillaume, J.L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. *J. Stat. Mech. Theory Exp.* **2008**, *2008*, P10008. [CrossRef]
84. Danon, L.; Díaz-Guilera, A.; Duch, J.; Arenas, A. Comparing community structure identification. *J. Stat. Mech. Theory Exp.* **2005**, *2005*, P09008. [CrossRef]
85. Hubert, L.; Arabic, P. Comparing Partitions. *J. Classif.* **1985**, *2*, 193–218. [CrossRef]
86. Meilă, M. Comparing clusterings–an information based distance. *J. Multivar. Anal.* **2007**, *98*, 873–895. [CrossRef]