Patient Engagement: Theoretical and Heuristic Approaches for Supporting the Clinical Practice.

Italo Zoppis[1], Riccardo Dondi[2], Sara Manzoni[1], and Giancarlo Mauri[1]

[1] Department Computer Science, University of Milano Bicocca, Milano, Italy
[2] Department of Letters, Philosophy, Communication, University of Bergamo, Bergamo, Italy

Abstract. Social interaction allows to support the disease management by creating online spaces where patients can interact with clinicians, and share experiences with other patients. Therefore, promoting targeted communication in online social spaces is a mean to group patients around shared goals, offer emotional support, and finally engage patients in their healthcare decision-making process. In this paper, we approach the argument from a theoretical perspective: we design an optimization problem aimed to encourage the creation of (induced) sub-networks of patients which, being recently diagnosed, wish to deepen the knowledge about their medical treatment with some other similar profiled patients, which have already been followed up by specific (even alternative) care centers. In particular, due to the computational hardness of the proposed problem, we provide approximated solutions based on distributed heuristics (i.e., Genetic Algorithms). Results are given for simulated data using Erdös-Rényi random graphs.

Keywords: Social Network · Distributed Heuristic · Genetic Algorithm · Dense Communities · 2-club optimization problem

1 Introduction

Social media can directly support the disease management by creating online spaces where patients can interact with clinicians, and share experiences with other patients [11,27]. For example, cancer patients use Twitter to discuss treatments and provide psychological support [32], and online engagement seems to correlate with lower levels of self reported stress and depression [7].

Similarly, wellness programs frequently incorporate social media to create a sense of community [34], group people around shared goals, and offer social and emotional support. A trial reported that adding on line community features to an Internet-mediated wellness and walking program improves adherence, and did reduce participant attrition [24].

Nevertheless, much more work remains to be carried out for sharing targeted and optimized information content. How can we optimize a procedure which is

able to facilitate the encounter between patients who want to deepen or share experiences about treatments, care points, and specialists? How to correlate, for example, similar clinical profiles, while inducing networks of medical stuff, and treated patients which offer their availability to share experiences or suggestions? These are exactly the issues we deal with in this paper.
In particular, we focus on the problem of creating a space of individuals and care centers, by considering the case where recently diagnosed patients could be interested to meet some other patients (experience), for sharing information on their disease and the suggested (or available) care center. In this situation, it would be useful, for example, to encourage the diagnosed subjects to socialize, and confront with the experience of other patients with similar clinical profile, who have been already followed up within the same (or even alternative) proposed care point.

It is clear that a proper handling of procedures and data is fundamental in order to convert available information into useful formulation that leads to particular (induced) communities. From a theoretical perspective this consideration can be expressed as the problem of finding dense or cohesive subgraphs (with particular properties), inside a network. Unfortunately, as we will report in the next sections, the intrinsic complexity of the considered computational problem make optimization potentially impracticable. For this reason, we design a distributed heuristic (i.e., Genetic Algorithms, GAs) to seek faster approximation solutions (see, e.g., [20] for details). In particular, we propose a general framework, which, similarly to a Map-Reduce programming paradigm, abstracts away the complexity of learning by allowing programmers to describe their processes in terms of two main procedures, here referred as *Training*, and *Combine*. With this approach we are able to provide both distributed concept learning and a proper delivering of the trained models on different nodes of the learning system.

In Section 2 we introduce the main theoretical aspects, focusing on the computational hardness of the considered problem (Sections 2.2). Then, we discuss the GA-based approach to seek approximated results for our formulation (Section 3). In Section 4, we extend this approach to a general distributed environment. Finally, after reporting numerical experiments on simulated data (Section 5), we conclude the paper (Section 6) by discussing our results and describing future directions of this research.

2 Main Definitions

Suppose we wish to model the situation where recently diagnosed patients (shortly reported as RD patients) are fostered to deepen the knowledge of their diseases with some other patient experience (say, engaged patients or shortly, ED), or they need more information concerning the suggested (or even alternative) health-care centers (HC). In this case, RD patients could benefit from the social interaction with similarly profiled (ED) patients, which have already been

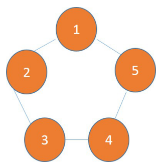

Fig. 1. An example of a 2-club consisting of 5 vertices, such that each subgraph of four vertices is not a 2-club. Assume that we remove vertex v_5; then the graph induced by $\{v_1, v_2, v_3, v_4\}$ is not a 2-club ($d_{G'}(v_1, v_4) = 3$).

followed up by specific HC centers. To this aim, it should be useful for a social platform to encourage, and optimize, the creation of a sub-network from the available social data. From a theoretical point of view, a network is most commonly modeled using a graph which represents relationships between objects, V (vertices), through a set of edges, E. In this way, our goal can be formulated by maximizing, within a defined graph, a *cohesive* sub-graph (i.e., by seeking the largest cohesive sub-graph) with particular properties (as we will detail in the following). Finding cohesive subgraphs inside a network is a well-known problem that has been applied in several contexts. For example, in computational biology, dense sub-graphs are sought in protein interaction networks, as they are considered related to protein complexes [4,31]. In gene networks, dense subgraphs are generally applied to detect relevant co-expression clusters [30]. While a classical approach to compute dense sub-graphs is the identification of cliques (i.e., complete sub-graphs induced by a set of vertices which are all pairwise connected by an edge), this definition is often too stringent for particular applications. This is the case, when the knowledge on how an individual (vertex) is embedded in the sub-network (e.g.,, some vertices could act as "bridges" between groups, as in our case) is a critical issue to take into account. Therefore alternative definitions of cohesive sub-graphs can be introduced, for example by relaxing some constraints, leading to the concept of *relaxed clique* [17]. Here, we follow this approach by relaxing the definition of distance between vertices. In a clique distinct vertices are at distance of 1, in our case, vertices can be at distance of at most $s = 2$. A sub-graph where all the vertices are at distance of at most 2 is called a 2-*club* (or, more in general, s-*club* for different values of s). 2-*clubs* have been extensively applied to social networks analysis [21,1,19,22], and biological network analysis, e.g., protein-protein interaction networks, [23]. Note that, when $s = 1$, a 1-club is exactly a clique. An example of 2-*club* with 5 vertices is represented in Figure 1.

2.1 Problem formulation

Consider a graph $G = (V, E)$, and a subset $V' \subseteq V$. We denote by $G[V']$ the subgraph of G induced by V'. Formally $G[V'] = (V', E')$, where

$$E' = \{\{u, v\} : u, v \in V' \wedge \{u, v\} \in E\}.$$

Given a set $V' \subseteq V$, we say that V' *induces* the graph $G[V']$[3].
The *distance* $d_G(u,v)$ between two vertices u, v of G, is the length of a shortest path in G which has u and v as endpoints. The *diameter* of a graph $G = (V, E)$ is $\max_{u,v \in V} d_G(u,v)$, i.e., the maximum distance between any two vertices of V. In other words, a 2-club in a graph $G = (V, E)$ is a sub-graph $G[W]$, with $W \subseteq V$, that has diameter of at most 2.
Moreover, given a vertex $v \in V$, we define the set $N(v)$ as follows

$$N(v) = \{u : \{v, u\} \in E\}$$

$N(v)$ is called the neighborhood of v. Finally, we denote by $N[v]$ the closed neighborhood of v, where $N[v] = N(v) \cup \{v\}$.

We will formulate the problem using 2-clubs whose shortest path connecting RD patients with specialist centers/staff (e.g., care center, hospital, or clinical staff) has to "transit" through, at least one EP patient who has already been followed up by the considered specialist "point".

Formally, for any pair, (d, h), composed by the recently diagnosed patient, d, and, e.g., the health care center, h, the social platform should suggest for the patient d, to compare (even to meet) with an identified (available) patient x's experience. More specifically, we are currently seeking (within the input "social graph") a 2-Club, $G[D \cup X \cup H]$, where D, X and H represent the sets of recently diagnosed patients, experienced patients, and care point centers. respectively. Please note that, when such a structure (i.e., a maximum size 2-clubs) exists, within the identified starting social space, then for any pair of vertices, it must exist at least one simple path of length 2, i.e., a path composed by a triple of vertices. This, in turn, will be also true for any pair, (d, h) where $d \in D, h \in H$. Indeed, our goal will be to find a largest-size 2-clubs which has the further property of providing, for any pair (d, h), a shortest path characterized by the triple of vertices $(d, x, h) \in (D \times X \times H)$. In this case, the set of edges, modeling the starting social network, will be defined as follow.

- Edges between similar profiled patients.
- Edges expressing that an experienced patient x, has already been followed up from the care center, h. In this case the edges in $X \times H$ will be constructed by knowing both the clinical history of each (experienced) patient, x, and the clinical staff or hospital h, which has already properly followed up the patients, x.
- Edges between vertices $h_1, h_2 \in H$, for example because two care centers are similar (have similar services or are part of the same institution).

In this situation, the simple path given by the triple of vertices $(d, x, h) \in (D \times X \times H)$ in the 2-club G would suggest for patient $d \in D$ to contact the patient, $x \in X$, about the health care center (or specialist staff), $h \in H$. For

[3] Notice that all the graphs we consider are undirected.

sake of clarity, before defining computationally the problem, we refer to any pair of vertices, $(d,h) \in D \times H$ (such that the minimum path connecting d to h is given by the vertex sequence (d,x,h), for any $x \in X$), as a "feasible pair". Considering the above discussion, we can define the following variant of the 2-clubs maximization problem.

Problem 1. Maximum 2-Club (Max-2-club)
Input: a graph $G = (D \cup X \cup H, R \cup F)$.
Output: a set $V' \subseteq D \cup X \cup H$ such that $G[V']$ is a 2-club having maximum size, and for each pair of vertices $(d,h) \in D \times H$ in $G[V']$, a minimum path connecting d to h is given by the vertex sequence (d,x,h) for some $x \in X$ (i.e., (d,h) is feasible).

2.2 Computational hardness

The complexity of the problem of Maximum s-club has been extensively studied in literature, and unfortunately it turns to be NP-hard for each $s \geq 1$ [9]; Maximum s-Club is NP-hard even if the input graph has diameter $s+1$, for each $s \geq 1$ [5]. The same property holds for our variant of Maximum 2-club. Indeed, the computation of a 2-club of maximum size containing a specific vertex v is also NP-hard. By defining $D = \{v\}$, $X = N(v)$ and H the remaining set of vertices, it follows that the "feasibility" property holds.
Given an input graph $G = (V,E)$, Maximum s-club is not approximable within factor $|V|^{1/2-\varepsilon}$, for any $\varepsilon > 0$ and $s \geq 2$ [3]. On the positive side, polynomial-time approximation algorithms [3] have been given, with factor $|V|^{1/2}$ for every even $s \geq 2$, and factor $|V|^{2/3}$ for every odd $s \geq 3$. The parametrized complexity of Maximum s-Club has also been studied, leading to fixed-parameter algorithms [28,18,10]. Maximum 2-Club has been considered also for specific graph classes [16,15].

3 A Genetic Algorithm

The complexity (and approximation) complexity of the problem introduced so far make optimization potentially impracticable. For this reason, we designed a Genetic Algorithm (GA) to seek faster approximation solutions see, e.g., [20] for details. In particular, given an input graph $G = (V,E)$, the GA represents a solution (a subset $V' \subseteq V$ such that $G[V']$ is a 2-club of G) as a binary chromosome c, of size $n = |V|$, such that for all $v_i \in V$, $c[i]$ is either 1 or 0. Note that, with a slight abuse of notation, we will denote by $G[c]$ the subgraph of G induced by the representation of chromosome c. Similarly, $V[c]$ and $E[c]$ will denote the set of vertices (V') and edges of $G[c] = G[V']$.
During the offspring generation, chromosomes are interpreted as hypotheses of feasible solutions (i.e., they may represent unfeasible solutions, for example an s-club, with $s > 2$, or a disconnected graph, during the first phases of evolution: they can later evolve in feasible solutions), undergoing to mutation, cross-validation and selection. Chromosome evaluation (i.e., hypothesis on a potential 2-club) is then provided, as usually, through the fitness function.

3.1 Fitness definition

In this paper, fitness is designed to promote adaptation in such a way that new candidate chromosomes, able to represent graphs with a feasible (correct) diameter value (i.e., not grater than 2) will "evolve" through the *elitism* mechanism. In this way, a small proportion of fittest candidates (chromosomes with high fitness values) are selected and preserved unchanged in the next generation. Specifically, given a chromosome c, and an input graph $G = (V, E)$, the fitness realizes such a mechanism using the following quantities.

1. An estimation of the number of the pairs of vertices, $(v_i, v_j) \in V[c] \times V[c]$, for which at least one shortest path (between v_i and v_j) of maximum distance 2 exists (in the considered chromosome representation $G[c]$.). For a consistent discussion, we will refer to such a collection of pairs (v_i, v_j) as the Feasible Set (FS), $\mathcal{C} \subseteq V[c] \times V[c]$.
2. The number of vertices, n_V, of the (sub)graph, $G[c]$, induced by c.

In particular, for any chromosome c, we observed a sample (of dimension n) $S \subseteq \mathcal{C}$, and by considering the induced subgraph representation $G[c]$, we evaluated the following fitness

$$f(n_v; \text{diam}) = \begin{cases} (1 - n_p/n_S)n_v & \text{if } 0 \leq \text{diam} \leq 2 \ ; \\ (1 - n_p/n_S)\frac{1}{n_v} & \text{if } 2 < \text{diam} \ , \end{cases} \tag{1}$$

where:

- n_v is the (set of vertices) cardinality of $G[c]$ induced (speculated) by c;
- n_p/n_S is the (simple *plug-in* sample) proportion

$$n^{-1} \sum_{k=1}^{n} I((v_i, v_j)_k \notin \mathcal{C}),$$

  of the ("unfeasibile") set of pairs, (v_i, v_j), whose shortest paths have dimension greater than 2, and
-

$$I(v_i, v_j)_k = \begin{cases} 1 & \text{if } (v_i, v_j)_k \notin \mathcal{C} \ ; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

In this way, by using the frequency, $(1 - n_p/n_S)$, of the sampled feasible pairs, the fitness weights (when $0 \leq \text{diam} \leq 2$) the number of vertecies n_v in $G[c]$, thus promoting large (sub)graph (size). On the other hand, given $\text{diam} > 2$ (i.e., unfeasible solutions) and $(1-n_p/n_S)$, we have fitness values which decrease asymptotically as $\frac{1}{n_v}$, for large (sub)graph size, n_v, thus penalizing the corresponding chromosome.

3.2 Search operators

To provide adaptation (with regard to the Maximum 2-club problem), we redefined the following standard operators.

- Mutation (applied with probability 0.75). Mutation must be carefully considered. Indeed, vertex mutations can affect not only the vertex involved in the current operations, but can even modify other parts of the solution. For this reason we defined two further mutation operators, both with the objective to correct hypotheses (i.e., chromosomes) consistently and parsimoniously. More specifically three type of mutations are considered.
  - Base Mutation (applied with probability 0.5, given that mutation has been chosen). Similarly to the standard case, each individual from the current population at time i is modified with a given probability (see details in the experimental section). In this case, mutation flips a bit of the selected chromosome c, in such a way that the corresponding vertex is either removed (i.e., bit flipped to 0) or added (i.e., bit flipped to 1) to the solution induced by c. Note that eliminating or adding vertices can induce subgraphs that actually do not have feasibility, i.e., they are not 2-clubs, since the property of being a 2-club is not hereditary. Moreover, modifications of chromosomes introduce the possibility of overcoming local minimum.
  - Non Standard Mutation 1 (applied with probability 0.25, given that mutation has been chosen). This modification has the objective to correct hypotheses (i.e., chromosomes) consistently and parsimoniously. Since any chromosome, by representation, induce a sub-graph $G[V']$ of G, which in turn may reflects feasible solutions, such hypotheses are partially verified using the following principle. Given a selected chromosome c, a vertex v' is (randomly) sampled from the set $V_+ = \{v_i : c[i] = 1\}$ and the minimum length of simple paths connecting every pair $(v_i, v'), v_i \in V_+$ is checked to be consistent with the chromosome representation, i.e., since each chromosome "speculates" a feasible 2-club, for such hypothesis to be true, there must be, at least, a simple path of size at most equal to 2 connecting any $v_i \in V_+$ with v'. If a negative feedback is observed after this verification, then the sampled vertex v' is flipped to 0.
  - Non Standard Mutation 2 (applied with probability 0.25, given that mutation has been chosen). This modification has the objective to increment (parsimoniously) the size of a solution. In this case, given a selected chromosome c a vertex v' is sampled from $V_- = \{v_j : c[j] = 0\}$ and the minimum length of simple paths connecting every pair (v_i, v') $(v_i \in V_+)$ is checked to be consistent with the current representation of the chromosome c. In this case, we consider to extend the hypothesis represented by c, by adding v' to V_+ if the minimum distances from v' to vertices of V_+ are not larger than 2.
- Cross-over (applied with probability 0.25). The following operations are provided.

    - Standard cross-over (applied with probability 0.5, given that crossover has been chosen). Offspring is generated by copying and mixing parts of parents' chromosomes. In this case a standard one-point crossover is implemented, and applied by the GA with probability 0.25.
    - Logical AND between parents (applied with probability 0.25, given that crossover has been chosen). This operation has the objective to provide an offspring consistent with the selected parents. For this, pairs of chromosomes are generated through logical AND operations between the ascendents. This operator i chosen with probability 0.5 (conditioned by the application of Logical Cross-over
    - Logical OR between parents (applied with probability 0.25, given that crossover has been chosen). This operation has the objective to provide offspring extending parent hypotheses. Extension is given by realizing a logical OR operation between two selected parents.
- Elitist selection (or elitism) In order to guarantee that solution quality does not decrease from one generation to another [6], best (25%) hypotheses (high fitness values) are allowed to be part of a new offspring.

3.3 Computational Cost of the Search Operators

Mutation and cross-over do not affect the computational cost of the GA evolution. As discussed above, given a selected chromosome c, a vertex v' is (randomly) sampled from $V_+ = \{v_i : c[i] = 1\}$ and the minimum length of simple paths connecting every pair $(v_i, v'), v_i \in V_+$ is checked to be consistent with the chromosome representation. For this, we intersect, for each $v_i \in V_+$, the closed neighborhoods $N[v_i]$ and $N[v']$. If an empty set is obtained (i.e., any vertex adjacent both to v_i and v' exists), then the corresponding i-th bit in v', is flipped to 0, this way, thus not increasing the complexity of the procedure. A similar argument applies when switching a bit from 0 to 1 (i.e., mutation 2; as defined above).

Conceptually, crossover is "easier" than mutation: it permits the generation of new chromosomes by implementing simple logical AND (respectively, OR) operations (as defined previously), which still do not affect, computationally, the process.

Finally, by observing the fitness formulation, we note that its complexity is mainly due to the diameter evaluation, which is known to be bounded by $O(|V|^3)$, thus preserving the tractability of the whole procedure.

4 Distributed learning

As reported above, the huge amount of data obtainable from social interaction activities, and the intrinsic complexity of the proposed computational problem make practically untreatable the optimization goal. An effective distribution of

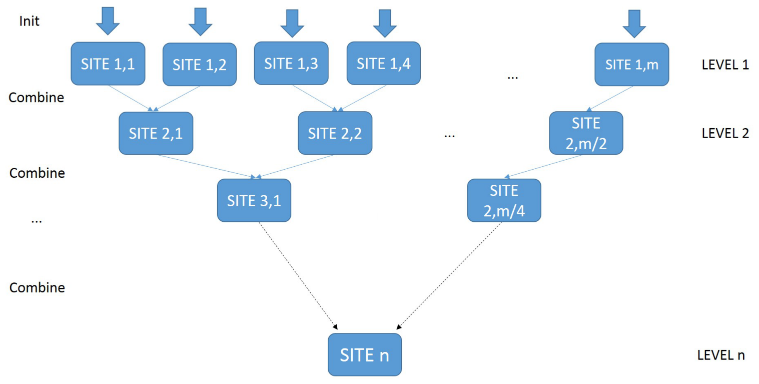

Fig. 2. The distributed GA enviroment: Data are "mapped" over different sites and best fitted chromosomes are combined into lower level population.

the computational load over different processors is therefore desirable.

In this section, we focus on how to obtain an efficient load distribution, by providing both parallel concept learning and a proper delivering of the trained (i.e., evolved) models on different nodes of a learning system. To this concern, we design a general framework, which applies the previous GA approach, and similarly to the Map Reduce programming paradigm, abstracts away the computational complexity of "learning" by allowing programmers to describe their processes in terms of two main procedures, which we will call *Training*, and *Combine*.

4.1 A genetic cascade model

Learning is practically based following the process represented in Figure 2. Different nodes (referred as "sites") realize, locally, the training phases by implementing the GAs described in Section 3. Beginning at the highest level, and progressing towards the lowest level, information on models, trained by the higher sites, are combined (equivalently, following a Map-Reduce "logic", models are "reduced") into sites of lower levels. The following functions are provided by the framework.

1. *Init.* To initialize both the training parameters and the number of first level sites. *Init* realizes the "data mapping" over the first level sites, i.e., it distributes the input data for the first training phase.
2. *Training.* Sites read local data as mapped by the Init function, and provides the first level models (i.e., best fitted chromosomes): GAs are executed by using the process described in the *Training* function. Similarly to an ensemble learning systems, sites are trained independently by applying the approach described previously.

3. *Combine*. Sites are used as filters. Following a boosting approach, the i-th level in Figure 2 attempts to create more accurate and general levels (i.e., $i+1$) of solutions by inserting into site $s_{i+1,k}$'s population (i.e., suggesting to site k of level $i+1$) the best chromosome (solutions) obtained by both "parents" $s_{i,k}$ and $s_{i,k+1}$. In other terms, a lower level population is initialized with parents' best chromosomes.

Large scale optimization is divided into independent, local smaller optimization problems, where best fitted chromosomes are distributed among different sites of the system. Best solutions are finally "reduced" (i.e., *Combined*) into the lowest level chromosome population, in a hierarchical fashion. Moreover, without any loss of generality, by assuming that patients and structures (i.e., input data and care centers) are initially ordered, each site provides local solutions (if any) on the corresponding input subnetwork. This process is detailed in Figure 3. In this case, since best fitted chromosomes from site $A1$ can clearly represent (speculate) feasible solutions from its input data only, one can obtain from site $A1$ a partial community (solution) referred to the associated input subnetwork $A1$, or equivalently corresponding to the respective adjacency (sub)matrix. Similarly, from site $A2$ input data, we can obtain a feasible local community for the corresponding subnetwork $A2$. The Combine operation extends the analysis to obtain feasible chromosomes for the subnetwork $B1$: local solutions represented by best chromosomes of site $A1$, and those from site $A2$ are combined and inserted into the new population of site $B1$. The new offspring and the obtained best chromosomes representation will provide optimized feasible solutions (if any) for the corresponding extended input data, which in turns represent a larger local solution suggested by the ("parents") $A1$ and $A2$ sites. Finally the lowest level site completes the evolution (for all the input data), starting from an optimized (suggested) population. Its solutions correspond (represent) to 2-club communities for the global input network.

5 Numerical Experiments

The genetic algorithm described in Sec. 3 was coded in R using the Genetic Algorithm ("igraph") package [29]. As for the cascade model, we used the standard "Multiprocessing" Pyhton package, working on local concurrency with 4 cores (i.e., the cascade model uses 4 starting sites). R and Python interfaces were managed by the rpy2 (`https://rpy2.readthedocs.io/en/version_2.8.x`) utility. Experiments are executed on Apple OSX 10.12.6; System Type MacBook Pro Retina; Processor: Intel(R), Core(TM) i5-6360U CPU @ 2.00GHz, 3.100 Ghz, 2 Core(s), 4 Logical Processors; Installed Physical Memory (RAM) 8,00 GB. Results are given for synthetic data, by sampling Erdos-Renyi random graphs $ER(n,p)$ with different values of number of vertices, n, and probability, p, to create edges between (pair of) vertices [8]. Numerical experiments have the main objective to evaluate both the effectiveness of the distributed learning and the capability of the GAs to obtain correct solutions in a reasonable time.

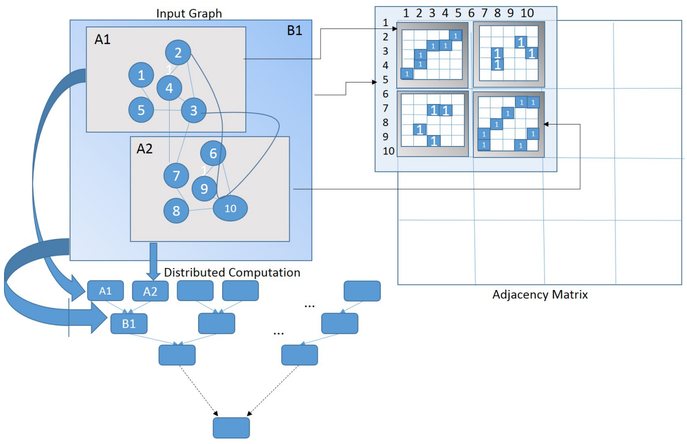

Fig. 3. Distributed GAs enviroment. Data are "mapped" over different sites and best fitted chromosomes combined in lower levels.

To provide correctness at "reasonable" cost, we followed the standard practice of the evolutionary algorithms: keep the tractability of the search operators and the fitness, and promote, at the same time, evaluable chromosomes, which in our case, provide feasible input diameter values. Moreover, a widely applied principle for termination has been applied: we set up both the number of re-evaluation of the fitness over new populations (equivalently, the number of the GA iterations), and the number of consecutive generations without improvement of the best fitness value[4]. Note that to check the robustness of the solutions, each ER model has been sampled iteratively 5 times. The corresponding standard deviation, obtained in each experiment, will be reported together with the obtained performances in the following section.

5.1 Results

The results are reported in Tables 1 and 2. In particular the following attributes are considered.

- Input and output diameters. The best GA solution is re-coded, and the resulting graph diameter is reported.
- Number of Output vertices. Input vertices are given by the first parameter, n, of the random model $ER(n, p)$
- Fitness value. Fitness is described in Sec. 3.
- Ratio between the number of the input graph vertices and the number of vertices of the resulting 2–club.

[4] See, e.g. [26], for a critical analysis of various aspects associated with the specification of termination conditions in a simple genetic algorithm.

Table 1. Models (Erdos-Renyi). Input Diameter (InD), Output Diameter (OutD), Output Nodes (OutN), Output Feasible Pairs (OutP).

Models	InD	OutD	OutN	OutP
Centralized Execution				
ER(150,0.1)	3.3 (0.48)	2 (0)	18.5 (3.53)	25.5 (21.2)
ER(150,0.2)	3 (0)	2 (0)	52.8 (38.3)	387.4 (633.6)
ER(500,0.1)	3 (0)	3.4 (3.3)	16.9 (9.6)	34.9 (36)
ER(500,0.2)	2 (0)	2 (0)	430.3 (68.5)	20840.9 (6164)
ER(1500,0.1)	2.43 (0.51)	2 (0)	105.4 (52.6)	1278.5 (983.3)
Distributed Execution				
ER(150,0.1)	3.2 (0.42)	2 (0)	18.8 (3.12)	28.9 (18.3)
ER(150,0.2)	3 (0)	2 (0)	53.3 (25.6)	294.6 (298.9)
ER(500,0.1)	3 (0)	2.6 (1.1)	42.6 (49)	183.5 (496.3)
ER(500,0.2)	2 (0)	2 (0)	346.6 (110.3)	14519.9 (9588.9)
ER(1500,0.1)	2.25 (0.46)	2 (0)	104.6 (22.7)	1180.8 (645.3)

Table 2. Models (Erdos-Renyi), Best Fitness (Fit), Iteration, Cpu Av. Time., Ratio between input and output Vertices.

Models	Fit	iter	Time	Ratio
Centralized Execution				
ER(150,0.1)	18.5 (3.53)	88.6 (14.9)	43.47 (14.9)	8.10
ER(150,0.2)	52.8 (38.3)	267.6 (147.3)	263.8 (120.1)	2.84
ER(500,0.1)	16.9 (9.6)	64.74 (27.32)	295.83 (163.4)	29.6
ER(500,0.2)	430.3 (68.5)	46.09 (70.86)	910.37 (1580.29)	1.16
ER(1500,0.1)	105.4 (52.6)	408.43 (235.08)	1171.79 (1056.6)	14.23
Distributed Execution				
ER(150,0.1)	18.8 (3.12)	20.2 (5.2)	47.50 (21.7)	7.97
ER(150,0.2)	53.3 (25.6)	59.8 (32.4)	77.56 (32.13)	2.81
ER(500,0.1)	42.6 (49)	13.29 (10.57)	119.8 (233.65)	11.73
ER(500,0.2)	346.6 (110.3)	4.33 (6.40)	107.89 (522.38)	1.44
ER(1500,0.1)	104.6 (22.7)	112.5 (10.35)	234.51 (247.24)	14.34

– Average CPU User Time per level in seconds. In case of a single processing unit, i.e., standard GA evolution, this quantity corresponds to the GA execution time. When the distributed process is considered this time is averaged by the number of framework levels.
– Early stopping for no improvement. The number of consecutive generations without improvement in the best fitness value before the GA is stopped.
– Max Number of Generation. The maximum number of iterations to run before the GA search is halted.
– Final generation number. Iteration number associated to the obtained solution. Note that for the distributed evolution this number corresponds to the iterations of the lowest site.

The following main considerations emerge from the results.

– In all model, execept $ER(500, 0.1)$, we obtained feasible 2-clubs (with null standard deviation).
– As for the considered network: we are able to find combinatorial structures which actually requires impractical computational time. In particular, this is interesting for those solutions which have been obtained for a large input graph (e.g., networks with more than 1000 vertices).

- It is clear that, due to the complexity of the problem, we cannot compare the size of the obtained 2-club community with the size of an optimal solution. Therefore, to give a qualitative idea of the solutions, we reported the ratio between the vertices of the input and the output graphs. Note that, the approximation complexity for the Maximum 2-club shows that the problem is not approximable within factor $|V|^{1/2-\varepsilon}$, for each $\varepsilon > 0$ [3]. Thus, even for the approximability, the problem is very hard to approach. Observing our results, and in particular the ratio in Table 2, we find that, the most compelling solutions are those for which, the larger is the input graph (number of vertices), the lower is the difference between the ratio and the unit.
- While the size of the inferred communities does not seem to differ, the (average) number of iterations of the last level site, for the distributed case, is much lower than the iterations reported by the standard, centralized evolution. This is also evident from the decrease in the average execution time per level, reported by the distributed case. Since GA uses the same parameters for the standard and the distributed evolution, this behavior can surely be traced back to the initial suggestions that each lower level site receives from the higher level.
- Computational cost seems to depend by the edge number (i.e., high expected connectivity of the random models $E(n,p)$) as well as the number of the input vertices. As discussed for the fitness 3.3, this assertion can be easily justified by considering the diameter computational cost, which is known to be bounded by $O(|V|^3)$.

6 Conclusions

While GAs optimization technique is not new in literature, a new design of these models is now needed to cope with the hardness of many computational problems, which actually find new applications in many contexts [13,14]. In this paper we focused on the optimization of a social network of patients which aim to deepen the knowledge about the available care centers for their pathologies through the help of other "experienced" patients. We considered this problem from a computational point of view by defining a variant of the max 2-club problem.
Due to computational complexity of the defined formulation, we provided GA-based heuristics on a distributed environment. The main advantage of this approach can be shortly summarized as follows.

- Similarly to the Map Reduce programming paradigm, the genetic cascade model described in Section 4.1 abstracts the computational complexity of learning by allowing programmers to describe the learning process in terms of *Training*, and *Combine* functions.
- Every site (as described in Section 4.1) provides local learning over the considered input graph. In fact, best fitted chromosomes generated by each GA's site represent local sub-graphs solutions. In this way, learning can be addressed to obtain feasible communities (i.e., 2-clubs) for specific issues: e.g.,

local optimized sub-networks characterized by particular (type of) patients or care centers.

- Large scale data optimization problems can be divided into independent, smaller optimization tasks. In this case, only local solutions are distributed downwards the system to create more accurate and general site level of computation.
- The proposed distributed learning environment is able to decrease the standard GA evolution time thanks to a series of appropriate initial suggestions, organized in a hierarchical manner.
- Finally, by considering a different learning algorithm, we can easily generalize the approach, simply by properly coding the *Training* function provided by the framework.

It is clear that, like in any problem solving paradigm, the question whether the considered approach reaches optimal solutions cannot be unconditionally positive expected. In fact, while the attempts to establish a theoretical account of GA's operations have proved more difficult in literature [33,12,25,2], the correctness of the solution is one of the desirable, yet not the only crucial issue (another is, for example, the time needed to reach correctness.).

Practice indicates that *self-adaptation* is a powerful tool to provide correct solutions: new chromosomes (i.e., solution representations) are generated by properly mutating, and/or recombining old solutions, and evaluable candidates (i.e., chromosomes) can survive (i.e., elitism) in a new population which, in turn, is immediately re-evaluated through the fitness function. Similarly, the number of re-evaluations is one of the most commonly used practical measure for assessing whether "efficiency", is (practically) achieved with reasonable cost. The approach given in this paper follows such a standard practice: while keeping the tractability of the search operators, i.e., mutation and crossover (as detailed in Sec. 3.3), the fitness (re)evaluation promotes correct populations by guaranteeing, in our case, feasible diameter values (as discussed in Sec. 3.1).

Many different extensions can be provided for this work. First of all, while a detailed comparison, between the presented approach and others comparable alternatives, is important for evaluating the practical conclusions of our paper, here, to the best of our knowledge, there are no state-of-the-art works to consider, for such a critical analysis. This question will be an issue which one should be consider for a further future (literature) research, in detail. Moreover, the distributed architecture reported in Section 4.1 was mainly introduced to help the combination and flow of information from higher level sites to lower level ones. In this paper, to speed up the experiments on laptops, we have used only a reduced number of levels and sites. Although this architecture is easily scalable to many different levels and sites, it is clear that, further architectures can be similarly applied. This is another interesting issue for a future research in this context.

Finally, it is important to emphasize that the work described in this paper has to be considered, as discussed in Section 1, a means to facilitate and promote the patient engagement. It is not absolutely intended as a constrain to the spontaneous nature of the communication and interaction activity, which characterize the freedom of a social networks.

References

1. R. D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3:113–126, 1973.
2. Lee Altenberg. The schema theorem and prices theorem. *Foundations of genetic algorithms*, 3:23–49, 1995.
3. Yuichi Asahiro, Eiji Miyano, and Kazuaki Samizo. Approximating maximum diameter-bounded subgraphs. In *LATIN 2010: Theoretical Informatics, 9th Latin American Symposium, Oaxaca, Mexico, April 19-23, 2010. Proceedings*, pages 615–626, 2010.
4. Gary D. Bader and Christopher W. V. Hogue. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics*, 4:2, 2003.
5. Balabhaskar Balasundaram, Sergiy Butenko, and Svyatoslav Trukhanov. Novel approaches for analyzing biological networks. *J. Comb. Optim.*, 10(1):23–39, 2005.
6. Shumeet Baluja and Rich Caruana. Removing the genetics from the standard genetic algorithm. In *Machine Learning, Proc. of the 12th Int. Conf. on Machine Learning, Tahoe City, California, USA*, pages 38–46, 1995.
7. Christopher E Beaudoin and Chen-Chao Tao. Modeling the impact of online cancer resources on supporters of cancer patients. *New Media & Society*, 10(2):321–344, 2008.
8. B. Bollobas. *Random Graphs.* Cambridge University Press, 2001.
9. Jean-Marie Bourjolly, Gilbert Laporte, and Gilles Pesant. An exact algorithm for the maximum k-club problem in an undirected graph. *European Journal of Operational Research*, 138(1):21–28, 2002.
10. Maw-Shang Chang, Ling-Ju Hung, Chih-Ren Lin, and Ping-Chen Su. Finding large k-clubs in undirected graphs. *Computing*, 95(9):739–758, 2013.
11. Enrico Coiera. Social networks, social media, and social diseases. *BMJ: British Medical Journal (Online)*, 346, 2013.
12. Pierre Collet and Jean-Philippe Rennard. Stochastic optimization algorithms. *arXiv preprint arXiv:0704.3780*, 2007.
13. R. Dondi, G. Mauri, and I. Zoppis. Orthology correction for gene tree reconstruction: Theoretical and experimental results. *Procedia Computer Science*, 108:1115–1124, 2017.
14. Riccardo Dondi, Giancarlo Mauri, and Italo Zoppis. Clique editing to support case versus control discrimination. In *Intelligent Decision Technologies 2016*, pages 27–36. Springer, 2016.
15. Petr A. Golovach, Pinar Heggernes, Dieter Kratsch, and Arash Rafiey. Finding clubs in graph classes. *Discrete Applied Mathematics*, 174:57–65, 2014.
16. Sepp Hartung, Christian Komusiewicz, and André Nichterlein. Parameterized algorithmics and computational experiments for finding 2-clubs. *J. Graph Algorithms Appl.*, 19(1):155–190, 2015.

17. Christian Komusiewicz. Multivariate algorithmics for finding cohesive subnetworks. *Algorithms*, 9(1):21, 2016.
18. Christian Komusiewicz and Manuel Sorge. An algorithmic framework for fixed-cardinality optimization in sparse graphs applied to dense subgraph problems. *Discrete Applied Mathematics*, 193:145–161, 2015.
19. Steven Laan, Maarten Marx, and Robert J. Mokken. Close communities in social networks: boroughs and 2-clubs. *Social Netw. Analys. Mining*, 6(1):20:1–20:16, 2016.
20. Melanie Mitchell. *An introduction to genetic algorithms.* Complex adaptive systems. MIT press, Cambridge (Mass.), 1996.
21. Robert Mokken. Cliques, clubs and clans. *Quality & Quantity: International Journal of Methodology*, 13(2):161–173, 1979.
22. Robert J. Mokken, Eelke M. Heemskerk, and Steven Laan. Close communication and 2-clubs in corporate networks: Europe 2010. *Social Netw. Analys. Mining*, 6(1):40:1–40:19, 2016.
23. Srinivas Pasupuleti. Detection of protein complexes in protein interaction networks using n-clubs. In *Evol. Comput., Machine Learning and Data Mining in Bioinf., 6th EvoBIO 2008, Naples, Italy, March 26-28, 2008*, pages 153–164, 2008.
24. Caroline R Richardson, Lorraine R Buis, Adrienne W Janney, David E Goodrich, Ananda Sen, Michael L Hess, Kathleen S Mehari, Laurie A Fortlage, Paul J Resnick, Brian J Zikmund-Fisher, et al. An online community improves adherence in an internet-mediated walking program. part 1: results of a randomized controlled trial. *J. of Med. Internet Res.*, 12(4), 2010.
25. Günter Rudolph. *Convergence properties of evolutionary algorithms.* Kovac, 1997.
26. Martín Safe, Jessica Carballido, Ignacio Ponzoni, and Nélida Brignole. On stopping criteria for genetic algorithms. *Advances in Artificial Intelligence–SBIA 2004*, pages 405–413, 2004.
27. Eugenio Santoro, Gianluca Castelnuovo, Italo Zoppis, Giancarlo Mauri, and Francesco Sicurello. Social media and mobile applications in chronic disease prevention and management. *Frontiers in psychology*, 6, 2015.
28. Alexander Schäfer, Christian Komusiewicz, Hannes Moser, and Rolf Niedermeier. Parameterized computational complexity of finding small-diameter subgraphs. *Optimization Letters*, 6(5):883–891, 2012.
29. L. Scrucca. GA: A package for genetic algorithms in R. *Journal of Statistical Software*, 53(4):1–37, 2013.
30. Roded Sharan and Ron Shamir. Center CLICK: A clustering algorithm with applications to gene expression analysis. In *Proc. of the Eighth Int. Conf. on Intelligent Syst. for Molecular Biol., August 19-23, 2000, La Jolla / San Diego, CA, USA*, pages 307–316, 2000.
31. V. Spirin and L. A. Mirny. Protein complexes and functional modules in molecular networks. *Proc. of the Nat. Academy of Sciences*, 100:12123–12–128, 2003.
32. Atsushi Tsuya, Yuya Sugawara, Atsushi Tanaka, and Hiroto Narimatsu. Do cancer patients tweet? examining the twitter use of cancer patients in japan. *Jour. of mMed. Internet research*, 16(5), 2014.
33. Thomas Weise, Michael Zapf, Raymond Chiong, and Antonio J Nebro. Why is optimization difficult? In *Nature-Inspired Algorithms for Optimisation*, pages 1–50. Springer, 2009.
34. Italo Zoppis, Giancarlo Mauri, Ferancesco Sicurello, Eugenio Santoro, Giada Pietrabissa, and Gianluca Castelnuovo. Diabesity: A study for mhealth integrated solutions. In *International Conference on Wireless Mobile Communication and Healthcare*, pages 195–199. Springer, 2016.