A Genetic Algorithm For Boolean Semiring Matrix Factorization With Applications To Graph Mining

Georgios Drakopoulos
Ionian University
0000-0002-0975-1877
c16drak@ionio.gr

Phivos Mylonas
Ionian University
0000-0002-6916-3129
fmylonas@ionio.gr

***Abstract*—Matrix factorization is paramount in large scale graph mining as well as a versatile paradigm for dimensionality reduction. In particular, factoring a graph adjacency matrix may well reveal, depending on the specific factor properties, higher order structure. The latter describes global graph properties better compared to first order connectivity patterns such as vertex degrees. The Boolean semiring factorization of an adjacency matrix yields a product of two smaller and sparser matrices where the former contains disjoint fundamental vertex subsets and the latter combinations thereof. Therefore, the first factor represents community structure and the second has the cross connections between them. In this way graph partitioning and dimensionality reduction are simultaneously achieved. Because of the nature of the Boolean semiring, most common linear algebraic solvers cannot be applied. Moreover, the exact factorization is NP hard. To address these limitations, a genetic algorithm has been developed with evolutionary operations tailored to heuristically compute said factorization which offers interpretability and a high parallelism potential. Besides graph mining the major applications of the Boolean semiring factorization include role mining in enterprise database and operating system realms, curriculum design, and graph flows under inflexible uniqueness constraints. The results obtained by applying the proposed genetic algorithm to synthetic graph benchmarks are very encouraging.**

***Index Terms*—Boolean semiring, dimensionality reduction, matrix factorization, genetic algorithm, graph mining, role mining**

I. Introduction

In an arguably highly interconnected world graphs can be found literally everywhere. With the advent of linked data analytics and the development of appropriate tools, which range from NoSQL databases like Neo4j [1] to distributed systems such as Apache Spark [2], it is not an exaggeration to say that the entire planet has obtained a cohesive digital nervous system. In this context the role of linked data processing becomes central. Such processing should take into account not only topology but also functionality. For instance, in the case of social networks connectivity patterns may correspond to friendships and functionality to messages or posts. Alternatively, in a logistics scenario, topology may denote available routes and functionality the associated capacities and costs.

Graph signal processing (GSP) is an emerging research field where graphs are two dimensional signals, expressed primarily through the respective adjacency or Laplacian matrices, and new knowledge is extracted by appropriately defined signal processing methods [3]. The main challenge comes from the irregular domain induced by graph topology, which implies that elementary signal processing operations must be redefined before any composite operations can be built on them. However, this also leads to a novel viewpoint on problems such as graph clustering, Fiedler value estimation, Laplacian approximation, and quick Fourier eigenexpansions. Recently, graph neural networks (GNNs) have emerged as a neural network architecture for distributed and recursive processing with emphasis placed on topology and communication. Because of the computational complexity of mining large graphs, for instance for adjacency matrix factorization or community structure discovery, efficient heuristics have been proposed.

Among the graph processing algorithms the class of graph partitioning stands out as it has numerous applications across various fields including social media analysis, logistics and long supply chains, protein interaction analysis, and omics data. As there is no single definition of what constitutes a community, there is a plethora of relevant algorithms. The Boolean semiring matrix factorization of an adjacency matrix yields sparse community structure. However, because of its definition linear algebraic techniques do not work and most proposed algorithms rely on nonlinear optimization principles.

TABLE I
Notation of this conference paper.

Symbol	Meaning	First in
$\triangleq$	Definition or equality by definition	Eq. (1)
$\vee$	Boolean disjunction (logical or)	Eq. (4)
$\wedge$	Boolean conjunction (logical and)	Eq. (4)
$\odot$	Matrix multiplication over semiring	Eq. (11)
$\oplus$	Boolean exclusive disjunction	Eq. (2)
$\{s_1, \ldots, s_n\}$	Set with elements $s_1, \ldots, s_n$	Eq. (1)
$\lvert\cdot\rvert$	Set cardinality functional	Eq. (17)
$\lVert\cdot\rVert$	Matrix or vector norm	Eq. (14)
$\mathrm{tr}\,(\cdot)$	Matrix trace	Eq. (14)
$\mathbf{e}_k$	k-th vector of standard basis	Eq. (18)
$\mathrm{E}\,[\mathcal{X}]$	Mean value of r.v. $\mathcal{X}$	Eq. (7)
$\mathrm{Var}\,[\mathcal{X}]$	Variance of r.v. $\mathcal{X}$	Eq. (7)
$\mathrm{prob}\,\{\Omega\}$	Probabilty of event Ω occurring	Eq. (6)

The primary research objective of this conference paper is the development of a genetic algorithm (GA) for selecting one of the two factors of the matrix factorization over the Boolean

978-1-6654-8045-1/22/$31.00 ©2022 IEEE

semiring, which is a well known NP-hard problem. The basic evolutionary operations have been tailored to the nature of this factorization. In turn, this allows for explainability and transparency in the way the proposed algorithm works.

The remainder of this work is structured as follows. In section II the recent scientific literature regarding graph mining, matrix factorization, and role mining is briefly overviewed. The proposed methodology including the genetic algorithm heuristic is described in detail in section III, whereas the results obtained are analyzed in section IV. Future research directions are given in section V. Boldface capital letters denote matrices, boldface lowercase vectors, and lowercase letters scalars. Acronyms are explained the first time they are encountered in text. Function parameters follow the respective arguments after a semicolon. Finally, the notation of this conference paper is summarized in table I.

II. Previous Work

Graph mining is a mainstay of linked data processing [4]. Graph partitioning [5] and subgraph mining [6] are problems closely related to graph community structure [7]. Role mining is an NP-hard problem [8]. In [9] is shown that role mining solutions rely on advanced non-linear optimization techniques. A library for role mining algorithms is proposed in [10]. Mining time dependent graphs is recently the focus of intense research [11]. Anomaly detection is a prime problem is these graphs [12]. Applications include searching for trusted candidates for startups on LinkedIn [13], XGBoost computation optimization [14], efficient GNN topology discovery [15], personalized recommendation for cultral content [16], ranking cloud providers [17], drug discovery [18], processing compressed graph sequences [19], graph autoencoders [20], and biomedical image clustering with tensor metrics [21].

GSP is a recently formulated field focusing on signal processing operations on graphs [3] treated as irregular domains [22]. Basic operations include graph sampling and graph filtering [23], while advanced ones are graph Laplace transform [24], graph spectral clustering in various forms [25] [26], and graph signal denoising [27]. GSP for incomplete topologies is the focus of [28]. The connections between GSP and ML are explored in [29]. The development of GNNs has further boosted GSP [30]. Architectures have been proposed for spam detection in social networks [31], for learning Markov processes [32], for evaluating the affective coherence of fuzzy [33] and ordinary [34] Twitter graphs, and for autoregressive moving average (ARMA) filtering [35]. Since graph patterns are typically of higher order, principles from higher order signal processing may well be applicable [36]. Moreover, explainability in GNNs is the focus of [37].

Matrix factorization in general has been proposed as a methodology for dimensionality reduction. The ubiquitous singular value decomposition (SVD) [38] appears in various applications such as determining the similarity of biomedical images [39], finding the spectral image of cyber attacks [40], and fuzzy versions for Kalman filters [41]. Polar decomposition [42], which is one of the major computationally feasible ways to obtain the singular value decomposition of a matrix [43], has been used among others for approximating directed graphs with undirected ones [44], for efficiently computing factored linear operators in quantum computing [45], and for evaluating the viability of capital market companies [46]. Nonnegative matrix factorization has been proposed for mining biological [47] and massive omic [48] data as well as for discovering community structure [49] and link prediction [50] in graphs, especially in conjunction with low rank matrix approximation [51]. Moreover, it can be employed for attribute extraction from ECG [52] and for image processing [53].

III. Proposed Methodology

A. Boolean Semiring

Informally speaking a semiring $\mathbb{S}$ is a well defined algebraic structure akin to a ring. However, there is a crucial difference in that each ring element has an additive inverse, whereas in a semiring at least one of its elements lacks such an inverse. The formal definition of a semiring is given in definition 1.

Definition 1 (Semiring)**.** *A semiring $\mathbb{S}$ over a field V having two binary operators $+$ and $\cdot$ satisfies the following conditions for every element of the underlying field $s, s', p, q \in V$:*

- *Additive neutral element:* $\exists s$ *s.t.* $\forall p\colon s + p = p$
- *Additive associativity:* $s + (p + q) = (s + p) + q$
- *Additive commutativity:* $p + q = q + p$
- *Multiplicative neutral element:* $\exists s'$ *s.t.* $\forall p\colon s' \cdot p = p$
- *Multiplicative associativity:* $s \cdot (p \cdot q) = (s \cdot p) \cdot q$
- *Left distributivity:* $s \cdot (p + q) = (s \cdot p) + (s \cdot q)$
- *Right distributivity:* $(p + q) \cdot s = (p \cdot s) + (q \cdot s)$

The operations of conjunction and disjunction over the set B create a semiring in the sense of definition 1, where B contains the two possible logical values as in (1). In this context and given their truth tables, disjunction and conjunction can be interpreted as Boolean addition and multiplication respectively.

$$B \triangleq \{0, 1\} \tag{1}$$

Definition 2 states that the Boolean semiring $\mathbb{B}$ is in fact a special semiring case over $\mathbb{B}$ as not only the requirements of definition 1 but also the additional multipliaction associativity condition holds as it can be readily verified from Boolean algebra. The semiring structure of $\mathbb{B}$ stems directly from the fact that the only way for the disjunction of two variables to be zero is when both of them to be zero. Otherwise, it suffices that only one operand be one. In other words, disjunction has no inverse element, although it has a neutral element.

Definition 2 (Boolean commutative semiring)**.** *The Boolean semiring $\mathbb{B}$ over B having the operations of disjunction as addition and conjunction as multiplication is a commutative semiring since it also holds true for every $p, q \in B$:*

- *Multiplicative commutativity:* $p \cdot q = q \cdot p, \quad \forall p, q$

As a sidenote, if exclusive disjunction (exclusive or) is used as the addition operation instead of disjunction over B, then a ring results as an additive inverse does exist for every element.

In fact, in this case each element of B is its own additive inverse as shown in equation (2):

$$s \oplus s \;=\; (s \wedge \neg s) \vee (\neg s \wedge s) \;=\; 0, \quad \forall s \in B \tag{2}$$

Equation (2) is tantamount to performing modular arithmetic with a base of two. This property is fundamental among others in coding theory for generating error correcting codes [54].

B. Inner Product Over The Boolean Semiring

A direct generalization of a single Boolean variable is a column vector as given in definition 3. Clearly such a vector of length n can take any of the 2^n possible values. Since matrices consist of stacked column vectors, a Boolean matrix can be similarly constructed. Also the transpose operator has analogous semantics with those in linear algebra, so Boolean row vectors require no special definition.

Definition 3 (Boolean vector)**.** *A Boolean column vector of length n over $\mathbb{B}$ consists of n variables, whether independent or not. Although each such variable can be independently accessed and modified, a vector is a single entity.*

$$\mathbf{b} \triangleq \begin{bmatrix} b_1, & \ldots, & b_n \end{bmatrix}^T \in \mathbb{B}^{n \times 1} \tag{3}$$

For any two given Boolean vectors $\mathbf{a}$ and $\mathbf{s}$ of length n their inner product over the Boolean semiring is defined as shown in equation (4). Observe the latter is a cluster concept or a disjunctive normal form (DNF) representing an instance of the 2SAT problem, albeit in a De Morgan inverted sense since 2SAT is typically cast in a conjnctive normal form (CNF) or Krom form. In sharp contrast to the general SAT or the k-SAT problems where $k \geq 3$ which are NP-complete, the 2SAT is NL-complete [55], meaning it is among the hardest problems requiring logarithmic space, and also belongs in P with a linear time solution based on implication graphs [56]. In fact, the sharp complexity change, termed the *phase transition*, from 2SAT to 3SAT is the focus of intense research [57].

$$\mathbf{a}^T\mathbf{s} \triangleq \bigvee_{k=1}^{n} (\mathbf{a}\,[k] \wedge \mathbf{s}\,[k]), \quad \mathbf{a}, \mathbf{s} \in \mathbb{B}^{n \times 1} \tag{4}$$

By construction it follows that the inner product of vectors $\mathbf{a}$ and $\mathbf{s}$ is true when there is at least one index k for which both $\mathbf{a}\,[k]$ and $\mathbf{s}\,[k]$ are true. In the deterministic case, the number of these pairs depends on the vector length as well as on their respective densities. The latter depends only on the nature of the underlying field and it can be only determined statistically. Also note that in the ensuing analysis the Boolean values of true and false are also interpreted numerically as one and zero respectively in the proper context. Thus for a Boolean vector, namely a column vector consisting exclusively of Boolean variables, the number of elements which are true equals the number of non-zero elements. Both expressions will be interchangeably throughout in this work.

A better understanding of the behavior of equation (4) can be achieved with by probabilistic analysis. Assuming that each element of both Boolean vectors is independently chosen to be true with probabilities p_a and p_s respectively and false with probabilities $1 - p_a$ and $1 - p_s$ again respectively, then each conjunction pair is a Bernoulli r.v. $\mathcal{I}_k$ with success probability $p_a p_s$ as shown in (5). Observe moreover that probabilities p_a and p_s essentially express the sparsity of the respective vectors.

$$\mathcal{I}_k \;=\; \begin{cases} \mathbf{a}\,[k] \wedge \mathbf{s}\,[k] = 1 & \text{with probability } p_a p_s \\ \text{otherwise} & \text{with probability } 1 - p_a p_s \end{cases} \tag{5}$$

Since each such conjunction pair contributes independently to the inner product of (4), the number of pairs evaluating to true is a r.v. $\mathcal{I}$ with the distribution of equation (6).

$$\text{prob}\,\{\mathcal{I} = i\} \;=\; \binom{n}{i} (p_a p_s)^i (1 - p_a p_s)^{n-i} \tag{6}$$

Observe that equation (6) is a binomial distribution with a fixed number of n trials and success probability $p_a p_s$, making it thus a member of the exponential family of distributions. Therefore, both p_a and p_s can be efficiently approximated with maximum likelihood estimators which additionally are unbiased and their variance attain the Cramér-Rao lower bound (CRLB). Moreover, from the from of the binomial distribution its mean and variance are established immediately as in (7).

$$\text{E}\,[\mathcal{I}] \;=\; np_a p_s \quad \text{and} \quad \text{Var}\,[\mathcal{I}] \;=\; np_a p_s (1 - p_a p_s) \tag{7}$$

The interpretation of (6) is that for even a moderate vector lengths there is sufficient probability that the inner product evaluates to true, especially when the success probability $p_a p_s$ approaches one. When the latter is sufficiently small, then the binomial distribution reduces to the Poisson distribution of (8):

$$\text{prob}\,\{\mathcal{I} \;=\; i\} \;=\; \frac{(np_a p_s)^i}{i!} e^{-np_a p_s} \tag{8}$$

This distribution approximation is derived as shown in equation (9), where each factor of (6) is separately approximated.

$$\binom{n}{i} \;\approx\; \frac{n^i}{i!} \quad \text{and} \quad (1 - p_a p_s)^{n-i} \;\approx\; e^{-np_a p_s} \tag{9}$$

In this case the expected value and variance are as shown in equation (10). Observe that now they are both essentially fractions of n with the same decay rate. The vanishing expected value of the Poisson distribution implies that the overall probability that (4) is true is small, whereas the equally vanishing distribution means there are actually few Boolean vector pairs whose inner product is true, or alternatively, that most dot products between any such pair taken at random is bound to be false with very high probability.

$$\text{E}\,[\mathcal{I}] \;=\; \text{Var}\,[\mathcal{I}] \;=\; np_a p_s \tag{10}$$

The interpretation of equation (8) is that since there are very few indices, as denoted by the small value of the product $p_a p_s$, where both $\mathbf{a}$ and $\mathbf{s}$ are true, then the probabilty of the inner product of (4) being true is exponentially small. This is aligned with the use of Poisson distribution as a model for rarely occurring events.

C. Boolean Semiring Matrix Factorization

In this subsection the matrix factorization problem over the Boolean semiring along with the primary constraints are described. Given the innder product of (4), factoring any data matrix $\mathbf{D} \in \mathbb{B}^{n\times n}$, including graph adjacency matrices, over the Boolean semiring $\mathbb{B}$ takes the form of equation (11):

$$\mathbf{D} \;=\; \mathbf{A} \odot \mathbf{S}, \quad \mathbf{A}, \mathbf{S}^T \in \mathbb{B}^{n\times d} \tag{11}$$

The structure of the *combination matrix* $\mathbf{A}$ and that of the *community matrix* $\mathbf{S}$ is given in equation (12). Observe that $\mathbf{A}$ is defined based on its rows, whereas $\mathbf{S}$ on its columns.

$$\mathbf{A} \triangleq \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_n \end{bmatrix} \quad \text{and} \quad \mathbf{S} \triangleq \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \dots & \mathbf{s}_n \end{bmatrix} \tag{12}$$

The matrix multiplication over the Boolean semiring used in (11) is defined as in (13), where the Boolean inner products between the rows and columns of the two matrix factors are defined as in equation (4). Observe that $\mathbf{A}$ is partitioned in rows, so there is no need for the transpose operator. Clearly this sort of matrix multiplication is of combinatorial nature.

$$\mathbf{A} \odot \mathbf{S} \triangleq \begin{bmatrix} \mathbf{a}_1\mathbf{s}_1 & \mathbf{a}_1\mathbf{s}_2 & \cdots & \mathbf{a}_1\mathbf{s}_n \\ \mathbf{a}_2\mathbf{s}_1 & \mathbf{a}_2\mathbf{s}_2 & \cdots & \mathbf{a}_2\mathbf{s}_n \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_n\mathbf{s}_1 & \mathbf{a}_n\mathbf{s}_2 & \cdots & \mathbf{a}_n\mathbf{s}_n \end{bmatrix} \in \mathbb{B}^{n\times n} \tag{13}$$

The approximate version of (11) assumes a cost function J_a in the form of equation (14). In this scenario matrices $\mathbf{A}$ and $\mathbf{S}$ are selected such that J_a is minimized under constraints such as computational resources or total response time.

$$J_a(\mathbf{A}, \mathbf{S}; \mathbf{D}) \triangleq \operatorname{argmin} \left[\|\mathbf{D} - \mathbf{A} \odot \mathbf{S}\|_F \right] \tag{14}$$

The Frobenius norm used in the cost definition of (14) is defined as in (15). The Frobenius norm is differentiable and invariable under unitary transforms, properties commonly employed in cost minimization algorithms. However, in this context neither can be used in the space spanned by $\mathbb{B}$ and, therefore, either advanced nonlinear optimization techniques or heuristics such as the proposed one can be used.

$$\|\mathbf{D} - \mathbf{A} \odot \mathbf{S}\|^2 \triangleq \operatorname{tr}\Big((\mathbf{D} - \mathbf{A} \odot \mathbf{S})^T (\mathbf{D} - \mathbf{A} \odot \mathbf{S}) \Big) \tag{15}$$

In addition to community discovery, both problems (11) and (14) can be cast as a dimensionality reduction problem since the resulting matrix factors $\mathbf{A}$ and $\mathbf{S}$ have a combined smaller number of elements compared to matrix $\mathbf{D}$. This is quantified by the density ρ_0 which is defined as the ratio of the total number of elements of $\mathbf{A}$ and $\mathbf{S}$ to these of $\mathbf{D}$ as in (16):

$$\rho_0 \triangleq \frac{dn + dn}{n^2} \;=\; \frac{2d}{n} \tag{16}$$

A figure of merit which is stricter than ρ_0 is J_s which is the total nonzero elements of $\mathbf{A}$ and $\mathbf{S}$ to these of $\mathbf{D}$.

$$J_s \triangleq \frac{|\{\mathbf{A}\,[i,j] \,|\, \mathbf{A}\,[i,j] = 1\} \cup \{\mathbf{S}\,[i,j] \,|\, \mathbf{S}\,[i,j] = 1\}|}{n^2} \;\leq\; \rho_0 \tag{17}$$

From the Boolean semiring definition it follows immediately that common linear algebraic operations such as matrix inversion, eigenexpansion, and linear system solution need to be redefined and, moreover, they may not be equally algorithmically tractable or even readily available as part of an efficient software library. This is one more reason for selecting a heuristic approach for solving the problem of (14).

D. Genetic Algorithm

The heuristic solution proposed here relies on the GA framework of algorithm 1. Therein the fundamental operations of selection, crossover, and mutation are defined as follows. The astute reader will observe that there is an additional loop. This happens because each candidate $\mathbf{S}_k$ is self-contained in the sense that it has all the vertices of the original graph. As such, each evolutionary operation should be applied only internally in each such candidate as, otherwise, evolutionary operations between two candidates may well yield a graph with a number of vertices considerably different than n.

Algorithm 1 Genetic algorithm framework.

Require: Adjacency matrix $\mathbf{D}$ and termination condition τ_0
Require: The parameters of the GA and hyperparameter η_0
Ensure: Matrix $\mathbf{D}$ is partitioned

```
obtain an estimate d* of d from algorithm 2
obtain a range r of admissible values around d*
for d_0 in r do
    create random population of N_s candidates S_k
    for N_g generations and while τ_0 is false do
        for all candidate partitionings S_k do
            select the fittest genes
            perform crossover
            if mutations allowed and mutation triggered then
                mutate a random gene
            end if
        end for
        obtain candidates A_k
    end for
end for
return best partitionings for each d_0 in r
```

For a given candidate $\mathbf{S}_k$ consists of n genes $s_{k,j}$ where $1 \leq j \leq n$. Initially each gene $s_{k,j}$, which represents the j-th column of the candidate $\mathbf{S}_k$, is randomly initiated. This is accomplished by assigning to each $s_{k,j}$ two things, one of the existing triangles of the original graph and a random number of the remaining vertices. Thus, initially each $s_{k,j}$ has at least one guarnateed community plus more than enough vertices to perform evolutionary search. This can be alternatively seen as each gene having been assigned a random partition of the columns of the identity matrix $\mathbf{I}_n$ as shown in (18):

$$s_{k,j} \triangleq \bigvee_{k'} \mathbf{e}_{k'}, \qquad k' \in \{1, \dots, n\} \tag{18}$$

Thus for a given $\mathbf{S}_k$ genes $s_{k,j}$ start as random communities and evolve towards more complex but structured ones. Thus,

essentially each candidate solution $\mathbf{S}_k$ remains sparse, encodes community structure with each column or gene $s_{k,j}$ being a separate community in the graph, and has the right dimensions. Moreover, knowing factor $\mathbf{S}$ in (11) means that $\mathbf{A}$ can be also determined by efficient algorithms proposed in the scientific literature [8] [9]. This representation avoids the potential exponential growth of the GA coding theorem.

Selection is perhaps the most straightforward to define. Specifically, the evaluation of each gene can be either done by assessing J_a for each product of $\mathbf{A}_k$ with $\mathbf{S}_k$. Notice that when such a product introduces a spurious edge, this is penalized by the very definition of J_a. Additionally, the respective sparsity J_s can be taken into consideration. The weighted harmonic mean of J_a and J_s as defined in equation (19) is ideal since it is small only when both arguments are small.

$$J(J_a, J_s; \eta_0) \triangleq \frac{1+\eta_0}{\frac{1}{\bar{J}_a} + \frac{\eta_0}{J_s}} = \frac{(1+\eta_0)\bar{J}_a J_s}{J_s + \eta_0 \bar{J}_a} \tag{19}$$

This sharply contrasts the arithmetic mean or average where an argument with a bigh numerical value can be compensated by a very small one yielding a small result. Thus, the harmonic mean strives to minimize both $\bar{J}_a$ and J_s simultaneously, subject to the general problem constraints and the specific instance dynamics. Moreover, it is less prone to numerical errors caused by decaying arguments. Please notice that the normalized approximation cost $\bar{J}_a$ is just J_a divided by n^2, namely its maximum possible value. This ensures that both $\bar{J}_a$ and J_s have the same range between zero and one.

Crossover consists of any two columns $s_{k,i}$ and $s_{k,j}$ exchanging vertices at random for the same $\mathbf{S}_k$. Mutation consists of randomly selecting two genes $s_{k,i}$ and $s_{k,j}$ and moving a single vertex from to the former to the latter. By design graph integrity restrictions regarding vertices are preserved. A failsafe policy is that the top N_b genes, namely those which have achieved the lowest penalties, are preserved intact in the gene pool and are moved to the next generation. Finally, the flow of the GA is shown in figure 1.

E. Dimension Selection

The only major question which needs to be addressed at this point is how the dimension estimation d^*, or equivalently the number of communities, is chosen. In general there is no definitive answer, as such insight into the structure of any graph is not known in advance. The core of the proposed approach consists in performing the power iteration clustering (PIC). In step 6 any clustering algorithm can be used.

Algorithm 2 may seem to defeat the purpose as a partitioning of $\mathbf{D}$ is indirectly obtained. Still, PIC operates on the assumption that $\mathbf{D}$ consists of continuous elements. Moreover, vectors $\mathbf{v}^{[k]}$ are computed with ordinary linear algebraic operations and not with those of the Boolean semiring. Thus, d^* is an approximation of the number of the communities.

Algorithm 2 Power iteration clustering.

Require: Matrix $\mathbf{D}$, termination condition τ_1
Ensure: Estimate d^* is obtained
1: select a random vector $\mathbf{v}^{[0]}$ of unit norm
2: **repeat**
3: compute $\mathbf{v}^{[k+1]}$ as $\mathbf{D}\mathbf{v}^{[k]}$
4: normalize $\mathbf{v}^{[k+1]}$ to unit norm
5: **until** condition τ_1 is **true**
6: cluster $\mathbf{v}^{[k+1]}$ and obtain number of clusters d^*
7: **return** d^*

IV. Results

A. Experimental Setup

In table II the experimental setup of this work is summarized. Therein the parameters mentioned in the previous sections are shown. Most of them can be tailored depending on the constraints of the underlying hardware.

TABLE II
Experimental Setup.

Parameter	Value
Number of runs R	1000
Graph size N_r	$[65536 : 16384 : 131072]$
Number of generations N_g	10000
Population size N_s	10000
Top gene size N_b	50
Mutation probability p_m	$1e-4$
Fitness function J	Eq. (19)
Hyperparameter η_0	1

The language of implementation was Julia, a relatively new but predominant choice for data driven and computationally intense applications. The latter are greatly faciliated by the rich functionality offered by the libraries, the inherent multithreading, the available data types, and the seamless scaling.

B. Evaluation

In figure 2 the relative number of generations as well as the relative wallclock time are shown with respect to the corresponding maximum values. Since the proposed GA is I/O bound, the total wallclock time is a very good approximation of the total computation time. There is clearly a positive trend, which can be attributed to the increasing size of the synthetic benchmark graphs and the associated increase of the size of the associated column spaces.

In figure 3 relative mean cost J of (19) with respect to its maximum value for the same 16 points of N_r were executed N_g times. The reason for doing the lattr is that the result of the proposed GA is stochastic and hence its expected value and variance are used. The parameter N_g is large enough to ensure statistical validity. For all sizes in the range of the experiments the cost was approximately constant and at acceptable levels as the maximum reconstruction error was 15.71%.

V. Conclusions And Future Work

This work casts graph partitioning as factoring the adjacency matrix over the Boolean semiring. The latter yields sparse par-

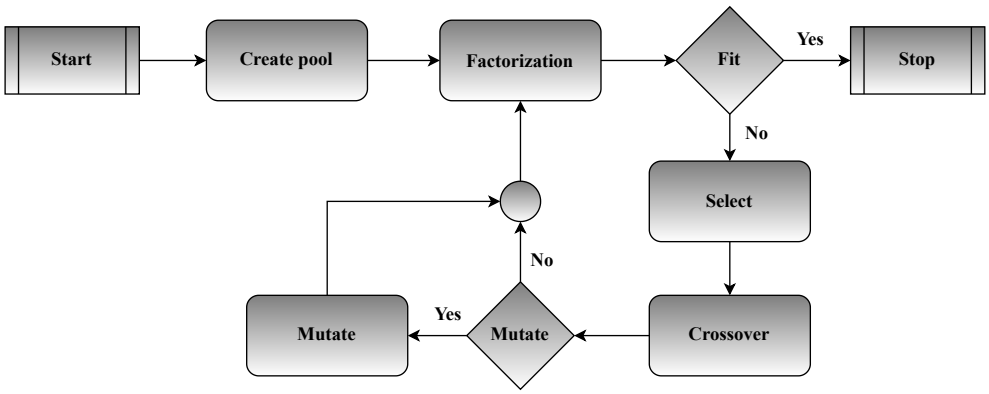

Fig. 1. Flowchart of the proposed genetic algorithm.

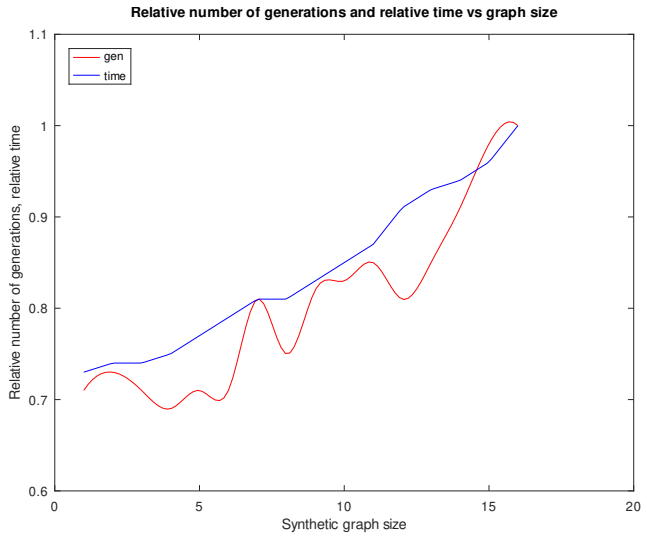

Fig. 2. Relative number of generations (max: 7556) and time (max: 96sec).

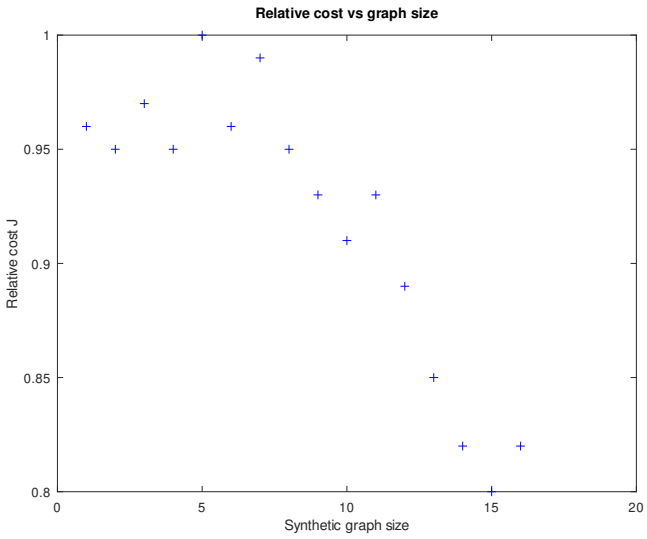

Fig. 3. Relative cost (max: 0.1571).

titions with no shared vertices, which is the classical definition of graph communities. However, because this factorization is NP-hard and moreover many linear algebraic operations differ, a genetic algorithm respecting vertex integrity was designed. Experiments with large benchmark graphs in Julia showed in efficiency, at least in the expected case. This work can be extended with more experiment with larger graphs. Additionally, different evolutionary operations can be designed or other heuristics can be tailored to the task.

ACKNOWLEDGMENT

This research was funded by the European Union and Greece (Partnership Agreement for the Development Framework 2014-2020) under the Regional Operational Programme Ionian Islands 2014-2020, project title: “Indirect costs for the project ‘TRaditional corfU Music PresErvation through digiTal innovation”’, project number: 5030952.

REFERENCES

[1] K. Rabuzin, M. Cerjan, and S. Križanić, “Supporting data types in Neo4j,” in *European Conference on Advances in Databases and Information Systems*. Springer, 2022, pp. 459–466.

[2] G. Cheng, S. Ying, B. Wang, and Y. Li, “Efficient performance prediction for Apache Spark,” *Journal of Parallel and Distributed Computing*, vol. 149, pp. 40–51, 2021.

[3] A. Ortega, *Introduction to graph signal processing*. Cambridge University Press, 2022.

[4] P. Yao, L. Zheng, Z. Zeng, Y. Huang, C. Gui, X. Liao, H. Jin, and J. Xue, “A locality-aware energy-efficient accelerator for graph mining applications,” in *MICRO*. IEEE/ACM, 2020, pp. 895–907.

[5] H. Xu, D. Luo, and L. Carin, “Scalable Gromov-Wasserstein learning for graph partitioning and matching,” *Advances in neural information processing systems*, vol. 32, 2019.

[6] L. B. Nguyen, I. Zelinka, V. Snasel, L. T. Nguyen, and B. Vo, “Subgraph mining in a large graph: A review,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2022.

[7] Z. Abbas, V. Kalavri, P. Carbone, and V. Vlassov, “Streaming graph partitioning: An experimental study,” *Proceedings of the VLDB Endowment*, vol. 11, no. 11, pp. 1590–1603, 2018.

[8] I. Molloy, N. Li, T. Li, Z. Mao, Q. Wang, and J. Lobo, “Evaluating role mining algorithms,” in *Symposium on access control models and technologies*. ACM, 2009, pp. 95–104.

[9] B. Mitra, S. Sural, J. Vaidya, and V. Atluri, “A survey of role mining,” *ACM CSUR*, vol. 48, no. 4, pp. 1–37, 2016.

[10] S. Anderer, B. Scheuermann, S. Mostaghim, P. Bauerle, and M. Beil, “RMPlib: A library of benchmarks for the role mining problem,” in *ACM Symposium on Access Control Models and Technologies*, 2021, pp. 3–13.

[11] Y. Wang, Y. Yuan, Y. Ma, and G. Wang, "Time-dependent graphs: Definitions, applications, and algorithms," *Data Science and Engineering*, vol. 4, no. 4, pp. 352–366, 2019.

[12] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, H. Xiong, and L. Akoglu, "A comprehensive survey on graph anomaly detection with deep learning," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[13] G. Drakopoulos, E. Kafeza, P. Mylonas, and H. Al Katheeri, "Building trusted startup teams from LinkedIn attributes: A higher order probabilistic analysis," in *ICTAI*. IEEE, 2020, pp. 867–874.

[14] J. Liu, S. Zhang, and H. Fan, "A two-stage hybrid credit risk prediction model based on XGBoost and graph-based deep neural network," *Expert Systems with Applications*, vol. 195, 2022.

[15] L. Wu, P. Cui, J. Pei, L. Zhao, and L. Song, "Graph neural networks," in *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer, 2022, pp. 27–37.

[16] G. Drakopoulos, I. Giannoukou, S. Sioutas, and P. Mylonas, "Self organizing maps for cultural content delivery," *NCAA*, 2022.

[17] T. E. Trueman, P. Narayanasamy, and J. Ashok Kumar, "A graph-based method for ranking of cloud service providers," *The Journal of Supercomputing*, vol. 78, no. 5, pp. 7260–7277, 2022.

[18] F. Zhong, X. Wu, R. Yang, X. Li, D. Wang, Z. Fu, X. Liu, X. Wan, T. Yang, Z. Fan *et al.*, "Drug target inference by mining transcriptional data using a novel graph convolutional network framework," *Protein & cell*, vol. 13, no. 4, pp. 281–301, 2022.

[19] G. Drakopoulos, E. Kafeza, P. Mylonas, and L. Iliadis, "Transform-based graph topology similarity metrics," *NCAA*, vol. 33, no. 23, pp. 16 363–16 375, 2021.

[20] Z. Sun, B. Wu, Y. Wang, and Y. Ye, "Sequential graph collaborative filtering," *Information Sciences*, vol. 592, pp. 244–260, 2022.

[21] G. Drakopoulos, I. Giannoukou, P. Mylonas, and S. Sioutas, "On tensor distances for self organizing maps: Clustering cognitive tasks," in *DEXA*, ser. Lecture Notes in Computer Science, vol. 12392. Springer, 2020, pp. 195–210.

[22] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[23] S. Liao and L. Shao, "Graph sampling based deep metric learning for generalizable person re-identification," in *CVPR*, 2022, pp. 7359–7368.

[24] H. Su, Y. Ye, X. Chen, and H. He, "Necessary and sufficient conditions for consensus in fractional-order multiagent systems via sampled data over directed graph," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 4, pp. 2501–2511, 2019.

[25] F. M. Bianchi, D. Grattarola, and C. Alippi, "Spectral clustering with graph neural networks for graph pooling," in *ICML*. PMLR, 2020, pp. 874–883.

[26] Y. Ding, Z. Zhang, X. Zhao, Y. Cai, S. Li, B. Deng, and W. Cai, "Self-supervised locality preserving low-pass graph convolutional embedding for large-scale hyperspectral image clustering," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–16, 2022.

[27] Y. Ma, X. Liu, T. Zhao, Y. Liu, J. Tang, and N. Shah, "A unified view on graph neural networks as graph signal denoising," in *CIKM*, 2021, pp. 1202–1211.

[28] E. Ceci and S. Barbarossa, "Graph signal processing in the presence of topology uncertainties," *IEEE Transactions on Signal Processing*, vol. 68, pp. 1558–1573, 2020.

[29] X. Dong, D. Thanou, L. Toni, M. Bronstein, and P. Frossard, "Graph signal processing for machine learning: A review and new perspectives," *IEEE Signal Processing Magazine*, vol. 37, no. 6, pp. 117–127, 2020.

[30] Y. Zhou, H. Zheng, X. Huang, S. Hao, D. Li, and J. Zhao, "Graph neural networks: Taxonomy, advances, and trends," *ACM TIST*, vol. 13, no. 1, pp. 1–54, 2022.

[31] Z. Guo, L. Tang, T. Guo, K. Yu, M. Alazab, and A. Shalaginov, "Deep graph neural network-based spammer detection under the perspective of heterogeneous cyberspace," *Future Generation Computer Systems*, vol. 117, pp. 205–218, 2021.

[32] Y. Lu, Y. Chen, D. Zhao, and D. Li, "MGRL: Graph neural network based inference in a Markov network with reinforcement learning for visual navigation," *Neurocomputing*, vol. 421, pp. 140–150, 2021.

[33] G. Drakopoulos, E. Kafeza, P. Mylonas, and S. Sioutas, "A graph neural network for fuzzy Twitter graphs," in *CIKM companion volume*, G. Cong and M. Ramanath, Eds., vol. 3052. CEUR-WS.org, 2021.

[34] G. Drakopoulos, I. Giannoukou, P. Mylonas, and S. Sioutas, "A graph neural network for assessing the affective coherence of Twitter graphs," in *IEEE Big Data*. IEEE, 2020, pp. 3618–3627.

[35] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi, "Graph neural networks with convolutional ARMA filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[36] A. Trapp and P. Wolfsteiner, "Estimating higher-order spectra via filtering-averaging," *Mechanical Systems and Signal Processing*, vol. 150, 2021.

[37] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," *IEEE TPAMI*, 2022.

[38] D. Broomhead, R. Jones, G. King, and E. Pike, "Singular system analysis with application to dynamical systems," in *Chaos, noise and fractals*. CRC Press, 2020, pp. 15–27.

[39] J. Baranger, B. Arnal, F. Perren, O. Baud, M. Tanter, and C. Demené, "Adaptive spatiotemporal SVD clutter filtering for ultrafast doppler imaging using similarity of spatial singular vectors," *IEEE Transactions on medical imaging*, vol. 37, no. 7, pp. 1574–1586, 2018.

[40] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," *Advances in neural information processing systems*, vol. 31, 2018.

[41] X. Yang, S. Wang, W. Xu, J. Qiao, C. Yu, and C. Fernandez, "Fuzzy adaptive singular value decomposition cubature Kalman filtering algorithm for lithium-ion battery state-of-charge estimation," *International Journal of Circuit Theory and Applications*, vol. 50, no. 2, pp. 614–632, 2022.

[42] R. Bru, M. T. Gassó, and M. Santana, "Combined matrices and conditioning," *Applied Mathematics and Computation*, vol. 412, 2022.

[43] P. Lv and B. Zheng, "Perturbation analysis for the QX factorization for centrosymmetric matrices," *Linear and Multilinear Algebra*, vol. 70, no. 3, pp. 557–580, 2022.

[44] G. Drakopoulos, E. Kafeza, P. Mylonas, and S. Sioutas, "Approximate high dimensional graph mining with matrix polar factorization: A Twitter application," in *IEEE Big Data*. IEEE, 2021, pp. 4441–4449.

[45] Y. Quek and P. Rebentrost, "Fast algorithm for quantum polar decomposition and applications," *Physical Review Research*, vol. 4, no. 1, 2022.

[46] F. Molaie Birgani, A. K. Salehi, M. Basirat, and A. Kaabomeir, "Providing a micmac analysis to strengthen sustainable green accounting values of capital market companies: Polar matrix analysis," *International Journal of Finance & Managerial Accounting*, vol. 7, no. 24, pp. 133–158, 2022.

[47] L. Ou-Yang, F. Lu, Z.-C. Zhang, and M. Wu, "Matrix factorization for biomedical link prediction and scRNA-seq data imputation: An empirical survey," *Briefings in Bioinformatics*, vol. 23, no. 1, 2022.

[48] A. R. Kriebel and J. D. Welch, "UINMF performs mosaic integration of single-cell multi-omic datasets using nonnegative matrix factorization," *Nature communications*, vol. 13, no. 1, pp. 1–17, 2022.

[49] Y. Zhao, F. Deng, J. Pei, and X. Yang, "Progressive deep non-negative matrix factorization architecture with graph convolution-based basis image reorganization," *Pattern Recognition*, vol. 132, 2022.

[50] E. Nasiri, K. Berahmand, and Y. Li, "Robust graph regularization non-negative matrix factorization for link prediction in attributed networks," *Multimedia Tools and Applications*, pp. 1–24, 2022.

[51] A. Agibetov, "Neural graph embeddings as explicit low-rank matrix factorization for link prediction," *Pattern Recognition*, vol. 133, 2022.

[52] Y. Huang, G. Yang, K. Wang, H. Liu, and Y. Yin, "Robust multi-feature collective non-negative matrix factorization for ecg biometrics," *Pattern Recognition*, vol. 123, 2022.

[53] T. Aonishi, R. Maruyama, T. Ito, H. Miyakawa, M. Murayama, and K. Ota, "Imaging data analysis using non-negative matrix factorization," *Neuroscience Research*, vol. 179, pp. 51–56, 2022.

[54] V. Berezhnoy, "Error correction method in modular redundant codes," in *International Conference on Mathematics and its Applications in new Computer Systems*. Springer, 2022, pp. 163–174.

[55] D. Achlioptas, A. Coja-Oghlan, M. Hahn-Klimroth, J. Lee, N. Müller, M. Penschuck, and G. Zhou, "The number of satisfying assignments of random 2-SAT formulas," *Random Structures & Algorithms*, vol. 58, no. 4, pp. 609–647, 2021.

[56] O. Omelchenko and A. A. Bulatov, "Satisfiability threshold for power law random 2-SAT in configuration model," *Theoretical Computer Science*, vol. 888, pp. 70–94, 2021.

[57] K. Richardson and A. Sabharwal, "Pushing the limits of rule reasoning in transformers through natural language satisfiability," in *AAAI*, vol. 36, no. 10, 2022, pp. 11 209–11 219.