Knowledge and Information Systems manuscript No.
(will be inserted by the editor)

A benchmarking tool for the generation of bipartite network models with overlapping communities

Alan Valejo · Fabiana Góes · Luzia Romanetto · Maria Cristina de Oliveira · Alneu de Andrade Lopes

Received: 24 Dec 2017 / Revised: 12 Jul 2019 / Accepted: 26 Aug 2019

Abstract Many real-world networks display hidden community structures with important potential implications in their dynamics. Many algorithms highly relevant to network analysis have been introduced to unveil community structures. Accurate assessment and comparison of alternative solutions are typically approached by benchmarking the target algorithm(s) on a set of diverse networks that exhibit a broad range of controlled features, ensuring the assessment contemplates multiple representative properties. Tools have been developed to synthesize bipartite networks, but none of the previous solutions address the issue of generating networks with overlapping community structures. This is the motivation for the BNOC tool introduced in this paper. It allows synthesizing bipartite networks that mimic a wide range of features from real-world networks, including overlapping community structures. Multiple parameters ensure flexibility in controlling the scale and topological properties of the networks and embedded communities. BNOC's applicability is illustrated assessing and comparing two popular overlapping community detection algorithms on bipartite networks, namely HLC and OSLOM. Results reveal interesting features of the algorithms in this scenario and confirm the relevant role played by a suitable benchmarking tool. Finally, to validate our approach, we present results comparing networks synthesized with BNOC with those obtained with an existing benchmarking tool and with already established sets of synthetic networks, in two different scenarios.

Keywords Complex Networks · Bipartite Networks · Heterogeneous Networks · Social Networks · Community Detection · Overlapping Community Detection · Benchmark · Synthetic Networks.

A. Valejo
E-mail: alanvalejo@gmail.com

Fabiana Góes
E-mail: fabianagoes@usp.br

Luzia Romanetto
E-mail: luzia@icmc.usp.br

Maria Cristina de Oliveira
E-mail: cristina@icmc.usp.br

Alneu de Andrade Lopes
E-mail: alneu@icmc.usp.br

Institute of Mathematics and Computer Science
University of São Paulo
P.O. Box 668, Zip Code 14560-970, São Carlos, SP, Brazil.

1 Introduction

The dynamics of many real-world systems is determined by the relations established between individual entities. This is the case, for example, of social networks, biological networks, information and technology networks [41]. Community structures in such complex networks are defined as a set of highly cohesive subsets of vertices and edges, also called components, partitions, or clusters. Such communities may be disjoint [21, 26, 53], i.e., each vertex belongs to a single community, or overlapping [46, 52, 51], when vertices may participate in multiple communities. The latter scenario is common in social networks, where individuals typically interact within multiple groups. Given a problem or a phenomenon described by a network model, vertices within a community typically share common characteristics or play similar roles.

Despite its relevance for characterizing a network's behavior, the community structure is unknown *a priori*. Yet, the analysis of network properties and dynamics demands a thorough understanding of its implicit topological organization. Community detection algorithms have been designed [21] to infer the actual organization of network vertices in a community structure. Their performance is typically measured in terms of the accuracy in identifying known representative and meaningful communities, combined with an analysis of computational complexity or execution times. Also relevant is how to accurately compare a novel algorithm with established ones. This is often accomplished by means of an empirical investigation conducted over a representative set of benchmark networks for which the community structure is known, so that results are amenable to validation and statistical analysis of the observed differences in performance. The benchmark networks should reflect the wide variety of features characterizing community structures, which may differ, e.g., in the number and size of communities, vertex degree distribution, presence of overlapping structures and also their extent and cohesiveness.

Benchmark networks representative of a diverse range of community organizations are necessary to disclose the limitations and advantages of the algorithms under investigation. However, obtaining ground-truth real-world networks exhibiting all the range of required features is hardly feasible, motivating the development of benchmarking tools to generate benchmark networks. The availability of such tools is paramount to support standardized, replicable and consistent empirical comparison of alternative algorithms [45]. A well-known contribution to the topic is the work by Lancichinetti et al [28]. Nonetheless, authors of a recent publication on the generation of hierarchical benchmark networks for community detection state that solutions contemplating specific scenarios are still lacking [54].

Bipartite networks, also known as two-layer networks, comprise a particular category of relational models in which the set of vertices is split into two disjoint sub-sets (the layers) and edges only occur between vertices in different layers. Let $G = (V, E, \omega)$ be an undirected weighted network, where $V = \{1, \ldots, n\}$ denotes the set of vertices, $E \subseteq V \times V$ is the set of edges, and ω is the set of weights assigned to edges. $|V|$ and $|E|$ denote, respectively, the number of elements in the vertex and in the edge sets. A network $G = (V, E, \omega)$ is bipartite if $V = V_0 \cup V_1$, such that $V_0 \cap V_1 = \emptyset$ and $E \subseteq V_0 \times V_1$.

This is a highly frequent class of networks that result from modeling pairwise relationships between entities of different types, e.g. documents and terms [20], patient and gene expression (or clinical variables) [25], individuals and songs (or books, or films) [24], scientific papers and their authors [39, 40]. Despite their widespread occurrence as models of real-world problems, the offer of benchmarking tools to assess community detection algorithms on bipartite networks is rather limited. Existing benchmarks are either not accessible to researchers or lack proper documentation. Indeed, the strong interest in novel algorithms to detect overlapped communities in bipartite networks is not matched by an offer of established benchmarking tools [19, 33, 34, 49, 16, 55].

In this paper we address this gap and introduce BNOC, a tool for synthesizing bipartite network models with varied features representative of properties from real networks. Multiple input parameters can be manipulated to create networks of varying sizes and with distinct community patterns in terms of number, size, balance, edge

distribution intra- and inter-communities, degree of overlapping and cohesion, and degree of noise in the connection patterns.

An empirical analysis varying the parameter settings confirmed the tool enables synthesizing community structures representative of a wide range of real-world bipartite networks. Our results show that BNOC can generate networks with hundreds of thousands vertices and hundreds of millions edges in acceptable times whilst demanding reasonable computing resources. The tool has been employed to benchmark HLC and OSLOM, two well-known overlapping community detection algorithms applicable to bipartite networks. Empirical results confirm its usefulness in uncovering the strengths and limitations of both algorithms. Finally, BNOC has been compared with an existing benchmarking tool and with already established sets of synthetic networks to assess its reliability.

The remainder of the paper is organized as follows: Section 2 reviews related work on generating benchmarking models; Section 3 introduces the tool, its implementation and resources; Section 4 reports results from an empirical study that illustrates its potential to create representative bipartite networks to benchmark community detection algorithms; Section 5 presents the concluding remarks; finally, Appendix A outlines the generalization of BNOC to address k-partite and heterogeneous networks.

2 Related Work

Vast collections of real-world networks are available from repositories such as the Stanford Large Network Dataset Collection[1], the Koblenz Network Collection[2], NetWiki[3], the UCI Network Data [4], the Newman Network Data[5] and Pajek[6]. However, the community organization of real networks is not known *a priori*, and accurate identification of outliers, overlapping vertices, community shapes and other potentially relevant features would demand extensive effort.

It is more feasible to rely on synthetic networks than to search for typical patterns in real-world problems, or take real networks as ground truth. A controlled variation of the relevant features of community structures embedded in a representative set of networks is an essential resource in establishing standardized and reusable procedures for algorithm assessment and comparison. Solutions exist to instantiate networks based on a particular underlying model, e.g. the scale-free network discussed by Barabasi and Bonabeau [7], or the scale-free network with bipartite structure introduced by Birmelé [11].

Papers that introduce novel community detection algorithms often report studies on synthetic networks, but such networks are seldom made available for reuse in future comparisons. It is possible to generate synthetic bipartite networks for benchmarking purposes with libraries such as Igraph[7] and NetworkX[8], e.g. fully connected, random and degree-based models, but they do not offer mechanisms to introduce arbitrary community structures.

There are benchmarks designed to assess community detection algorithms [37, 30], as well as tools for building realistic benchmarks of on-line social networks [9, 44, 5, 43, 42, 38]. Recent studies of large-scale social networks have resorted to simulated models [44, 6, 12, 13], e.g. the GN (Girvan and Newman) model [23] has been widely employed in this context. A popular, accessible and documented benchmark, known as the LFR model [28, 27]

[1] https://snap.stanford.edu/data/
[2] http://konect.uni-koblenz.de/
[3] http://netwiki.amath.unc.edu/SharedData/SharedData
[4] https://networkdata.ics.uci.edu/
[5] http://www-personal.umich.edu/~mejn/netdata/
[6] http://vlado.fmf.uni-lj.si/pub/networks/data/
[7] https://igraph.org
[8] https://networkx.github.io/

generates networks following a power-law vertex degree distribution [14, 3]. The LFR model has been recently extended to construct hierarchical networks for purposes of benchmarking community detection algorithms [54]. Alessandro and Vittorio [4] have discussed synthetic benchmarks to compare community detection and link prediction methods.

To the best of our knowledge, no existing tool addresses the specific problem of creating arbitrary bipartite networks with overlapping community structure for benchmarking purposes. There are contributions in the literature that describe usages of synthetic bipartite networks, e.g., [31] relied on a random model with added noise to create two sets of synthetic bipartite networks with disjoint communities to assess a stochastic block model community detection algorithm. Similarly, Barber [8] and Beckett [10] generated weighted bipartite networks with disjoint communities and Chakrabarti et al [14] synthesized directed and weighted bipartite networks. A model by Melamed [36] relies on density and disjoint community settings to create four categories of synthetic bipartite networks.

Availability of general bipartite networks for benchmarking, however, remains limited. Previous models are not necessarily accessible for reuse in later studies, or the information disclosed may be insufficient to ensure correct replication of the target networks. In fact, previous contributions do not address the problem of synthesizing reproducible bipartite network models with varying topological properties. Scripts, as provided, e.g. by Larremore et al [31] and Melamed [36], allow reproducing their models in similar experiments but were not conceived as documented general-purpose tools to synthesize a variety of reproducible bipartite networks. Moreover, these previous contributions did not address the generation of bipartite networks with overlapping communities.

This scenario motivated the development of a general-purpose benchmark generator of bipartite networks that ensures standardization and reproducibility in assessing community detection and other important algorithms.

3 A Tool for Benchmark Generation

In this section we introduce the BNOC benchmarking tool for synthesizing weighted **b**ipartite **n**etworks with **o**verlapping **c**ommunity structures. It can be employed to create networks with balanced or unbalanced overlapping communities, heterogeneous community sizes, intra- and inter-community edge density with varying average degrees and clustering coefficients. Table 1 lists the parameters that can be manipulated to create small or large-scale bipartite networks with distinctive features.

BNOC relies on the negative binomial distribution, defined in Equation 1, a discrete probability distribution that represents the number of possible failures in a sequence of Bernoulli trials prior to reaching a target number of successes.

$$BN(N; n, s) = \binom{N+n-1}{n-1} s^n (1-s)^n \tag{1}$$

In the previous equation $N + n - 1$ is the number of trials, $n - 1$ is the number of successes, and s is the probability of success. Therefore, the negative binomial distribution gives the probability of $n - 1$ successes and N failures in $N + n - 1$ trials with a success in the $(N + n)^{th}$ trial.

A negative binomial distribution can be generated as a gamma mixed Poisson distribution with $s = {}^{d}/_{d+mu}$ as gamma scale parameter, where d is the target number of successful trials, or dispersion parameter; and mu is a weighting factor that affects the shape of the distribution. Both parameters are strictly positive, i.e. d and $mu \in \mathbb{R}_+$. If parameter n in the probability distribution takes a non-integer value, the resulting probability is zero. Figure 1 illustrates the behavior of the negative binomial distribution for distinct choices of parameters mu and d, where $k = 10$ is the number of trials ($k = N + n - 1$).

Table 1 Description of the controlling parameters in BNOC.

Parameter	Type	Domain	Default	Description
-v, --vertices	$\|v\| = 2$	$[1, \|V\|] \subseteq \mathbb{Z}$	[10, 20]	Number of vertices for each layer
-c, --communities	$\|c\| = 2$	$[1, \|V\|] \subseteq \mathbb{Z}$	[2, 2]	Number of communities for each layer
-p0, --probability0	$\|p^0\| = c[0]$	$(0, \|V\|] \subseteq \mathbb{R}$	[0.3, 0.7]	Probabilities for vertices in layer 0 for each community
-p1, --probability1	$\|p^1\| = c[1]$	$(0, \|V\|] \subseteq \mathbb{R}$	[0.3, 0.7]	Probabilities for vertices in layer 1 for each community
-b, --balanced	boolean	$\{0, 1\}$	0	Boolean balancing flag that suppresses -p parameter
-x, --overlap	number	$[0, \|V_0|] \subseteq \mathbb{Z}$	1	Number of overlapping vertices in V_0
-y, --overlap	number	$[0, \|V_1|] \subseteq \mathbb{Z}$	1	Number of overlapping vertices in V_1
-z, --noverlap	number	$[0, \|c|] \subseteq \mathbb{Z}$	2	Number of overlapping communities
-d, --dispersion	number	$\mathbb{R}_+$	0.1	Dispersion of negative binomial distribution
-s, --success	number	$\mathbb{R}_+$	1	Probability of success
-n, --noise	number	$(0, 1] \subseteq \mathbb{R}$	0.01	Noise
-l, --normalize	boolean	$\{0, 1\}$	0	0-1 scale weight individually to unit norm
-u, --unweighted	boolean	$\{0, 1\}$	0	Unweighted bipartite networks

Small dispersion values ($d \approx 0$) result in strongly concentrated probability distributions approaching zero values, hence, likely to yield more failures. Otherwise, fewer failures occur as d increases, as the concentration towards null values is not as pronounced. This behavior is useful in generating synthetic networks.

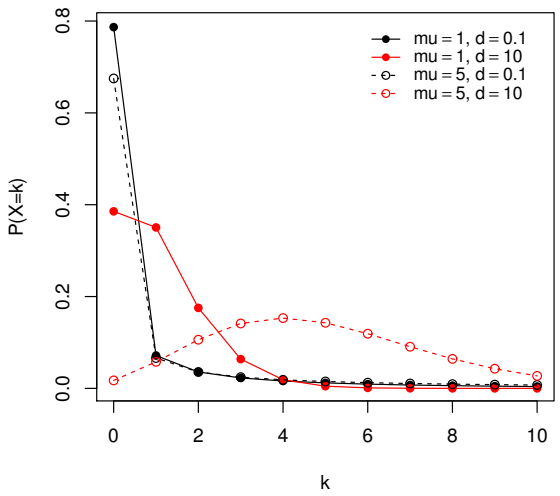


Fig. 1 Negative binomial distribution for distinct values of parameters *mu* and d.

For this purpose, we associate the probability of a success with the probability of a connection (i.e., an edge) being established between two vertices, whereas a failure indicates no connection. Model construction departs from an initial bipartite network fully connected and unweighted. The negative binomial distribution is employed to compute the edge weights, wherein null probability values to compute the edge weights, wherein null probability values cause edges to be eliminated, and non-null probability values are assigned as weights of the corresponding

preserved edges. Finally, it is possible to insert noise into the connection patterns initially established by the binomial distribution.

Usually networks are created departing from an empty edge set, to which edges and corresponding weights are added one by one, iteratively, while considering some restriction, e.g., satisfying a desired degree distribution. The process typically requires several steps of edge rewiring, as part of connections considered may violate the imposed restriction [21]. This increases the computational cost of the process. We adopt a more effective strategy, departing from a fully connected network with no weights and fitting a binomial distribution, which requires no iterative rewiring of undesired connections (though some rewiring is still required to establish noisy connections, see Step 5 below). The full procedure comprises the following steps:

Step 1: Creating vertices and communities.

Two initial input parameters define the desired number of vertices in each layer (array $v = [v_1, v_2]$, with $v_1 = v_2$ or $v_1 \neq v_2$) and the number of communities (c). The relative size of each community is determined from a probability distribution $p^0 = \{p^0_1, ..., p^0_c\}\ \forall\ p^0_i \in (0, 1] \subseteq \mathbb{R}$ and $p^1 = \{p^1_1, ..., p^1_c\}\ \forall\ p^1_i \in (0, 1] \subseteq \mathbb{R}$ for layers 0 and 1. Thus, a probability distribution p^i_j determines the relative size of community j in layer i. Higher probabilities will produce larger communities and, unless communities overlap, the sum of all community sizes must match the total number of vertices $|V|$. Alternatively, parameters p^0 and p^1 may be suppressed by a single parameter b if the aim is to produce a balanced community structure, i.e., equal-sized communities. Furthermore, vertices must be assigned to communities: initially all vertices are created, then an iterative process assigns each one to a community randomly chosen with probability p^0 or p^1. Iteration stops once all vertices have been assigned.

Step 2: Creating overlapping structures.

Three optional input parameters x, y, and z control community overlapping. Parameters x and y control, respectively, how many vertices from V_0 and V_1 will be assigned to multiple communities. Parameter z defines the maximum number of communities each overlapping vertex will be assigned to. Overlapping vertices are selected randomly, defining a set V' with $|V'| = x + y$. Similarly, overlapping communities are selected randomly, according to their respective probabilities p^i_j as defined in the previous step. Each vertex $v \in V'$ is assigned to up to z randomly chosen communities, with probability p^0 or p^1. Higher values of x, y will result in complex community structures more challenging to detect. These parameters are inspired by, and play a similar role, to those adopted in the widely adopted benchmarking solution by [21], which builds overlapping structures in one-layer networks.

Step 3: Connecting vertices within communities.

Vertices in the different layers that have been assigned to the same community are connected by an edge, so that all components become fully connected. Overlapping vertices are connected to all other vertices in their assigned overlapped communities, which produces components with overlapping regions.

Step 4: Setting edge weights and network density.

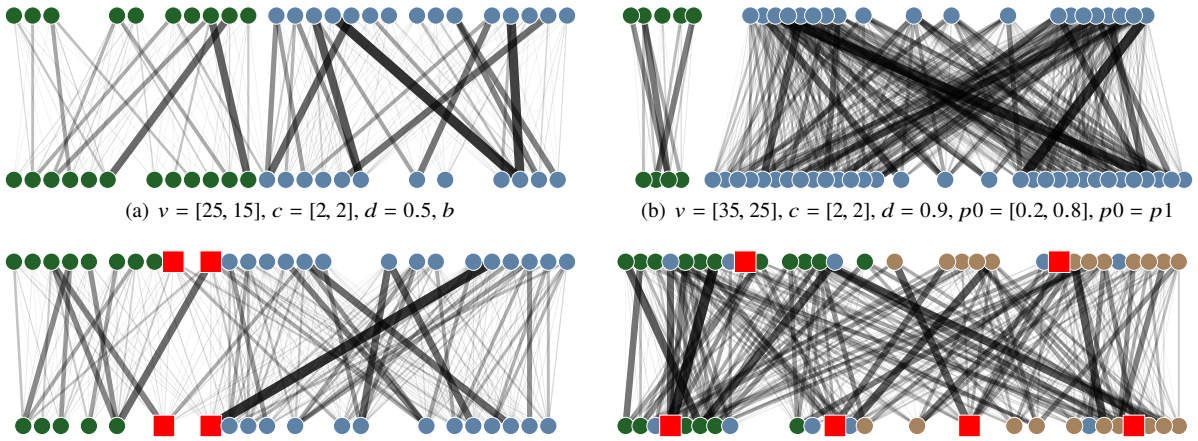
Each edge is assigned a weight obtained from the negative binomial distribution controlled by the dispersion parameter $d \in \mathbb{R}_+$ and success probability parameter $s \in \mathbb{R}_+$, which can be set to control community density locally. Low values of d and s yield a distribution with many null values, hence causing more edges to be removed. Therefore, lower values of dispersion and success rate ($d \approx 0$ and $s \approx 0$) yield sparser communities, whereas higher values ($d \approx 1$ and $s \approx 1$) yield denser communities. Alternatively, parameter l may be set to scale the weight vectors individually to the unit norm (vector length), and parameter u may be set to create unweighted networks, i.e., the only role of the distribution is to control edge sparsity.

Step 5: Adding inter-community edges and noise.

After executing Step 4 each vertex has been connected to other vertices in its own community. An optional parameter n enables inserting noise into the connection patterns at this point, so that the connections of randomly selected vertices will be rewired to vertices in other communities. Parameter $n \in (0, 1] \subseteq \mathbb{R}$ induces a rewiring of a proportion of intra-community edges, which are removed and replaced by randomly inserted inter-community edges, e.g. $n = 0.5$ causes around 50% of the edges to be rewired. Low values of n imply in inserting few inter-community edges, yielding networks with clear and easily separable communities. As n increases, more inter-community edges are inserted, increasing overlap and causing community boundaries to become blurrier. In general terms, $n > 0.5$ yields networks with poorly defined and sparse community structures depicting more inter-community than intra-community edges. Choices of $n \in [0.01, 0.4]$ are often adequate to increase the difficulty of the community detection task without drastically harming the community structure.

It is worth mentioning that BNOC creates a community organization following the concept of communities as cohesive groups of “densely" connected vertices sparsely connected to vertices outside the group. The “well-behaved” and balanced initial structure created in Step 1 can be modified in later steps by introducing noise to connection patterns or increasing community overlapping. Other benchmarking tools, e.g. LFR, adopt a similar approach. Nonetheless, a precise formulation of what constitutes a community structure remains elusive and other tools may adopt different perspectives. Rosvall et al [47] describe a categorization of community detection algorithms and emphasize there is no general-purpose community detection algorithm, as applications and data types may differ in their perspective of what characterizes a community structure.

Figure 2 illustrates multiple networks obtained with a few parameter combinations (non-default parameter values are informed at the bottom of each plot, default values are as indicated in Table 1).



(a) $v = [25, 15]$, $c = [2, 2]$, $d = 0.5$, b

(b) $v = [35, 25]$, $c = [2, 2]$, $d = 0.9$, $p0 = [0.2, 0.8]$, $p0 = p1$

(c) $v = [20, 25]$, $c = [2, 2]$, $p0 = [0.4, 0.6]$, $p0 = p1$, $x = 2$, $y = 2$, $z = 2$, $d = 0.6$

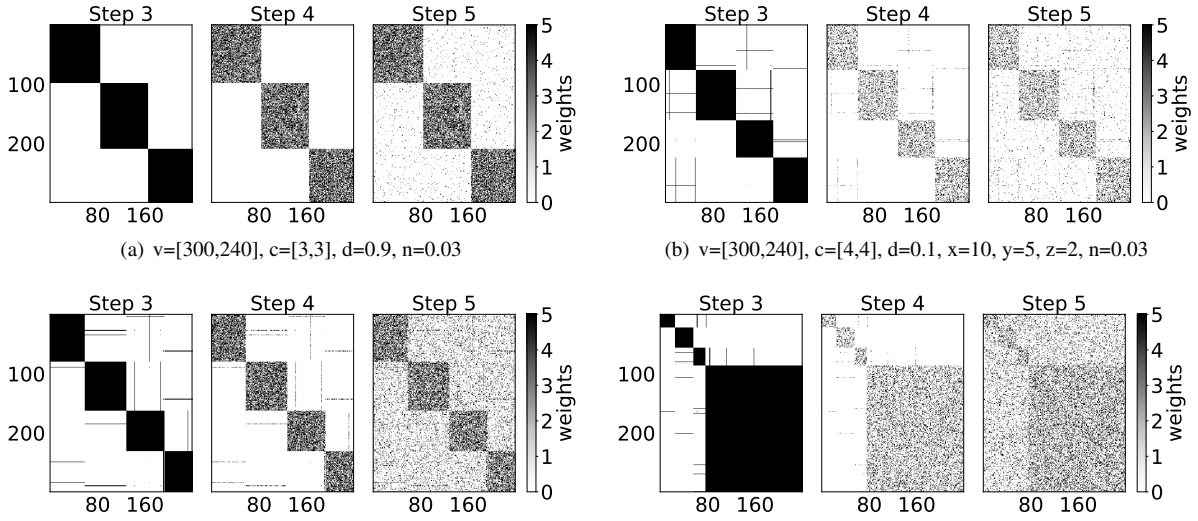
(d) $v = [25, 30]$, $c = [2, 2]$, $d = 0.8$, $x = 4$, $y = 2$, $z = 2$, $n = 0.2$, b

Fig. 2 Bipartite networks with distinct topological properties generated by BNOC. Line widths reflect the corresponding edge weights; red squares depict overlapping vertices and colored circles indicate non-overlapping vertices and their assigned community.

The network in Figure 2(a) exhibits a sparse topological structure, whereas Figure 2(b) illustrates a dense and unbalanced network. The network in Figure 2(c) is also dense and unbalanced, but it includes overlapping vertices. Finally, Figure 2(d) depicts a network with dense topology, extensive noise and overlapping vertices.

BNOC can be used to synthesize networks at varying levels of complexity that can be taken as good approximations of real-world models, which are often noisy and not necessarily "well-behaved". Such networks have blurred community boundaries and complex structures that may hinder algorithm performance and affect the community properties controlled in the aforementioned Steps 3 to 5. Their adjacency matrices, illustrated in Figure 3, show how a network's topological properties vary as different parameter settings are employed when executing Steps 3, 4 and 5.

Figure 3 illustrates four synthetic networks constructed executing Steps 3, 4 and 5 with different parameterizations. In Figure 3(a), a network is initially created with a dense, balanced and non-overlapping community structure (Step 3); then, edge weights are created and some edges are removed (Step 4); finally, noise is added to the connection patterns (Step 5). In Figure 3(b), the initial network with a balanced and overlapping community structure becomes sparser after executing step 4 and adding some noise (step 5). In Figure 3(c) an initial network with a dense, balanced, and overlapping community structure becomes sparser with extensive noise. Finally, in Figure 3(d) a network with an unbalanced overlapping community structure is modified to become highly sparse with extensive noise.



(a) v=[300,240], c=[3,3], d=0.9, n=0.03

(b) v=[300,240], c=[4,4], d=0.1, x=10, y=5, z=2, n=0.03

(c) v=[300,240], c=[4,4], d=0.9, x=10, y=10, z=3, n=0.2

(d) v=[300,240], p0=[0.1,0.1,0.1,0.7], p0=p1, c=[4,4], d=0.1, x=10, y=5, z=3, n=0.2

Fig. 3 Views of the adjacency matrices depicting how the topological properties of an initial network created in Step 3 are modified as Steps 4 and 5 are executed. Four scenarios are illustrated corresponding to alternative parameter settings: (a) an initial dense, balanced and non-overlapping community structure is modified by adding weights (causing some edges to be removed) and inserting limited noise; (b) a sparse, balanced, and overlapping community structure is modified with added weights and limited noise; (c) a dense, balanced, and overlapping community structure is modified with added weights and extensive noise; (d) a highly sparse, unbalanced, and overlapping community structure is modified with added weights and extensive noise.

A few topological features are indirectly controlled by a combination of certain parameter choices. For instance, network average degrees and local clustering coefficients depend on parameters c (number of communities), p^0 and p^1 (the probability distributions which define the size of each layer), d (dispersion) and s (probability of success). Denser bipartite networks resulting from higher dispersion values naturally yield higher average degrees and clustering coefficients.

Figure 4 illustrates the behavior of the average degree [32] for networks of distinct sizes, as a function of parameter d assuming $s = 1$ fixed and a balanced layer structure (p^0 and p^1 suppressed by setting parameter b). Lower dispersion values ($d \approx 0$) yield lower average degrees (between 3 and 10), whereas higher dispersion values ($d \approx 1$) yield high degree networks.

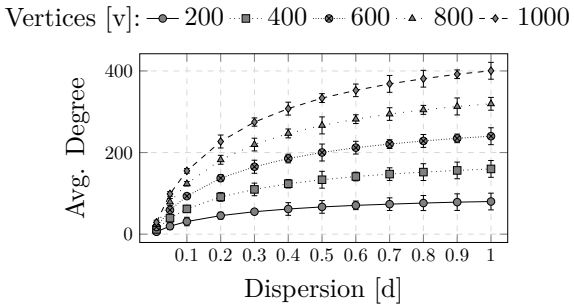


Fig. 4 Behavior of a network's average degree as a function of the dispersion parameter d.

Figure 5 shows the behavior of the average clustering coefficient [32] on four networks of distinct sizes, again as a function of dispersion parameter d. Low values result in networks with lower average clustering coefficients (between 0.1 and 0.2), whereas high values yield high clustering coefficients.

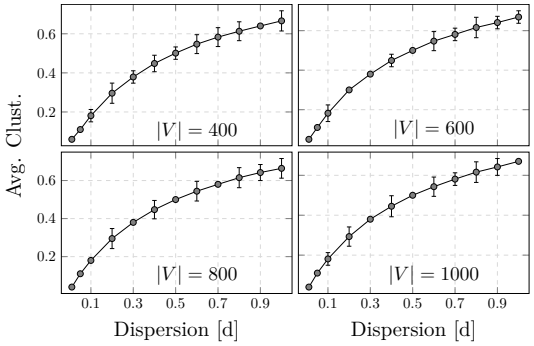
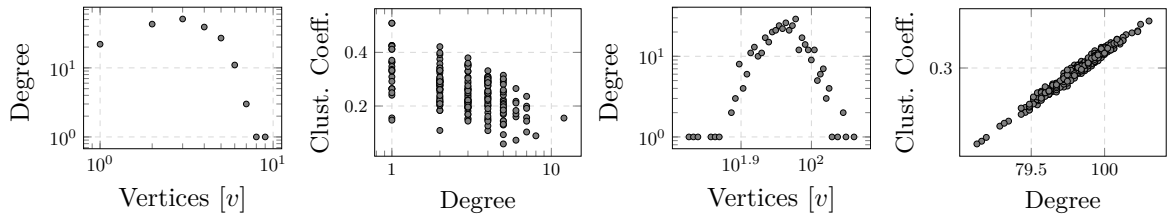


Fig. 5 Behavior of a network's average clustering coefficient as a function of dispersion parameter d, considering four network sizes (shown in separate graphs to avoid line overlapping).

Appropriate parameterization of BNOC´s underlying model enables simulating a wide range of behaviors representative of real-world networks. Figure 6(a) (top) shows the distribution of degree (left) and clustering coefficient (right) of a synthetic bipartite network with a few high-degree vertices and many low-degree. Figure 6(b) (bottom) illustrates the same information for a network with no community structure, obtained setting parameter $c = [1, 1]$, which leads to a topological structure close to a random model. According to Newman [39], in social networks vertex clustering coefficients and degree are typically inversely proportional.

Sparsity can be controlled by varying the edge distributions intra- and inter-community. In general, document-term networks [20] are sparse, whereas biological patient-gene expression (or patient-clinical) networks are characterized by dense structures [25]. The ability to independently control the size of each network layer allows

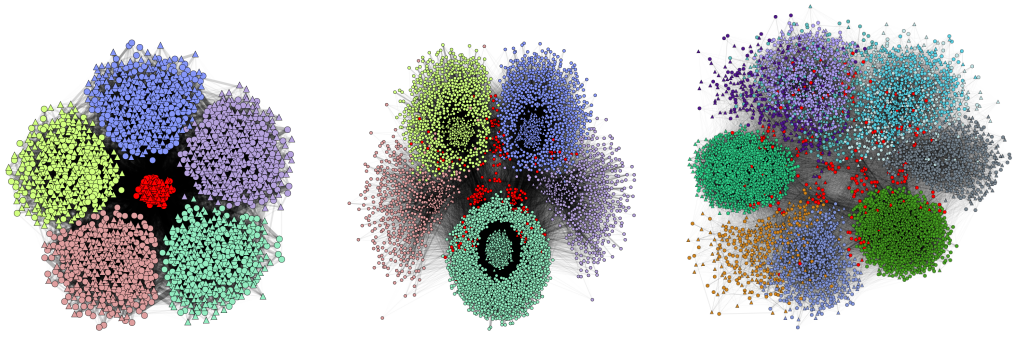
modeling a topological property of many real networks, e.g. in biological patient-gene expression networks there are many more genes than patients. Likewise, document-term networks typically have many more terms than documents. Again, in order to reflect properties of real networks, community structure and degree of vertex overlapping are also controlled separately for each layer. The extent of noise is controlled independently of the overlapping structure, since it is important to observe if a community detection algorithm can distinguish between vertex overlapping, i.e., vertices that belong to multiple communities, and noise.



(a) $v = [100, 100]$, $d = 0.1$, $c = [10, 10]$, $n = 0.1$, b (b) $v = [200, 200]$, $d = 0.2$, $c = [1, 1]$, b

Fig. 6 Distribution of vertex degree and local clustering coefficient of two distinct bipartite networks obtained with BNOC.

Overlapping vertices are simultaneously assigned to up to z communities randomly chosen considering the individual community probabilities p_j^i, as described in Step 2. Therefore, there will be vertices simultaneously assigned to exactly z communities and others assigned to less than z communities. Parameter z allows creating overlapping structures of varying complexity: high values, i.e., $z \approx c$, yield to strong, explicit and easily detectable community overlapping, especially for high values of c, since the overlapping vertices will belong to many communities simultaneously. In contrast, low values, i.e., $z \approx 2$, yield to more subtle overlapping, since the overlapping vertices will simultaneously belong to just a few communities. Figure 7 illustrates the effect of varying parameter z in a synthetic bipartite network initially with five communities in both layers.



(a) (b) (c)

Fig. 7 Bipartite networks generated with BNOC illustrating the effect of varying parameter z, which controls the degree of community overlapping. Red markers depict overlapping vertices, whereas other colors indicate the assigned communities of the non-overlapping vertices. (a) a network built with five communities ($c = [5, 5]$) and $z = 5$ (strong overlapping); (b) a network built with $c = [5, 5]$, $z = 3$, and $x = y = 80$; (c) a network built with $c = [10, 10]$, $z = 2$, and $x = y = 150$.

The red vertices are the overlapping ones, the remaining colors identify the vertices assigned to each of the distinct communities. Figure 7(a) exhibits the community structure of the network obtained with parameter settings $c = [5, 5]$, $z = 5$, $x = y = 50$; Figure 7(b) shows the community structure obtained with $c = [5, 5]$, $z = 3$ and $x = y = 80$; and Figure 7(c) results from setting $c = [10, 10]$, $z = 2$ and $x = y = 150$.

Specific parameters allow controlling the range and scale of edge weights to simulate the behavior of many real weighted or unweighted networks. Possible examples are user-movie networks such as IMDb (https://www.imdb.com/), in which registered users assign to films grades from 1 to 10; user and web-page networks, in which users may access a web-page (as Facebook) hundreds of times and others just a few times (as an on-line banking application) (both weighted networks); document-term networks, in which edges indicate the frequency of a term in a document (weights normalized to scale 0-1, obtained setting parameter l); or unweighted location-based on-line social networks with geographic check-ins (obtained setting parameter u).

Figure 8 illustrates a BNOC-generated bipartite network that presents some of the aforementioned distinct behaviors at each layer, such as sparsity, weight scale, size and community structure derived with different probability distributions.

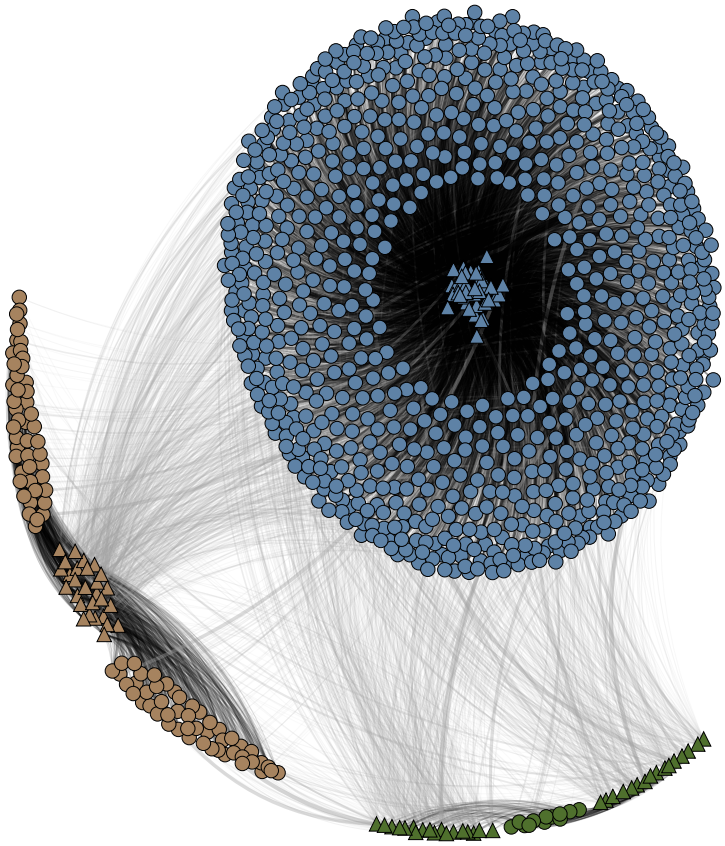


Fig. 8 Bipartite network generated with BNOC that illustrates properties of a hypothetical document-term network with 1, 000 terms (shown as circles) and 200 documents (shown as triangles) and three unbalanced communities. It has a sparse edge distribution, with normalized 0-1 weights. Globally the network has many more terms than documents, but this is not necessarily true of all communities: both the larger community in blue and the smaller community in brown have more terms than documents, unlike the smallest community, in green.

It might represent, for instance, a hypothetical corpus of political opinions about a specific topic expressed in three types of media documents such as political discourses, tweets and journalistic news, modeled as a bipartite document-term network [20, 2]. Globally, the term layer is considerably larger than the document layer, however, this global property does not necessarily hold locally in all communities. Let us say, it is possible that a community formed by the tweets might have more documents than terms (e.g., the green community in Figure 8), unlike the communities by the political discourses or media news (the brown and blue communities), or yet one of the communities might be balanced in the number of documents and terms.

Other complex community organizations could be considered in such a network, e.g., communities might model topics defined by multiple words, pronominal choices (or textual context), political parties or others. In each scenario, an investigation on how to parameterize the model to obtain the desired behavior would be required. While some features are directly controlled by individual parameter settings, others are determined by random or indirect choices.

Parameters d and s can be chosen in combination to generate sparse ($|E| \approx |V|$) or dense networks ($|E| \gg |V|$). Figure 9 shows curves of the number of edges (sparsity) as a function of the dispersion parameter d, for fixed values of the probability of success parameter s.

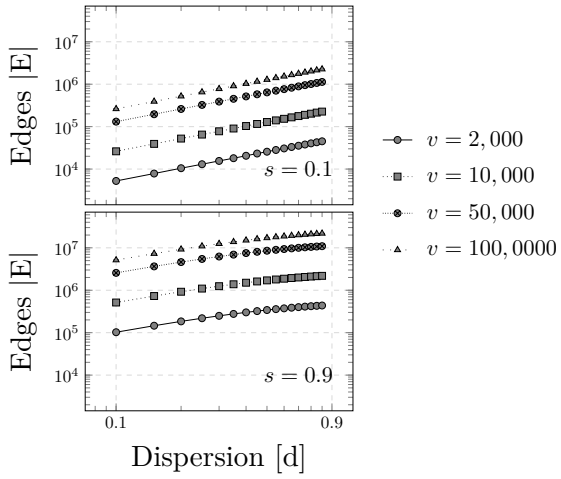


Fig. 9 Number of edges (sparsity) as a function of the dispersion parameter d.

Considering vertex sets of sizes 2,000 and 100,000, respectively, picking e.g., $d = 0.1$ and $s = 0.1$, yielded sparse networks with 5,000 and 200,000 edges, respectively; on the other hand, picking $d = 0.9$ and $s = 0.9$ generated dense networks with 400,000 and 25,000,000 edges, respectively.

We assessed the scalability of BNOC to generate large networks. Networks were created with parameter v varying in the range $[2,000; 100,000]$ at increments of 2,000, setting the number of communities $c = 0.01v$, dispersion $d = 0.1$, and probability of success $s = 0.9$. The curves depicting the times to create the networks as a function of the number of vertices are shown in Figure 10.

On sparse networks ($d = 0.1$ and $s = 0.1$), BNOC spent, in average, 0.2 and 19 seconds to build networks with $v = 2,000$ and $v = 100,000$ vertices, respectively. On dense networks ($d = 0.9$ and $s = 0.9$) it spent, in average, 0.3 and 20 seconds to build networks with $v = 2,000$ and $v = 100,000$ vertices, respectively. Therefore, execution times increase with the number of vertices, but sparsity does not impact the tool's scalability.

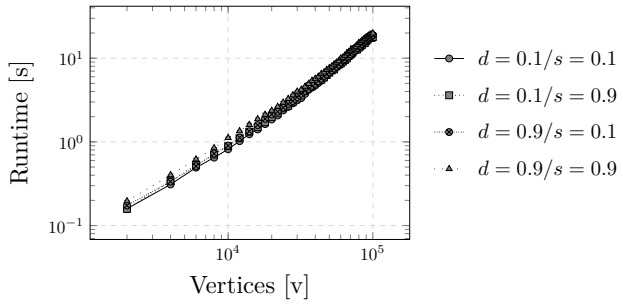


Fig. 10 Runtime as a function of the number of vertices v.

As opposite, the number of communities directly impacts scalability. Since edges are created to connect vertices within the same community, a small number of large communities implies more edges, and many communities implies less edges. Networks were generated with $v = \ \ 2,000$, $10,000$, $50,000$ and $100,000$ vertices setting the number of communities c within the range $[2, 160]$ at increments of 2, and $d = s = 0.5$. Resulting execution times are shown in Figure 11. For large-scale networks, with $100,000$ vertices, BNOC spent, in average, 643 seconds (or 10 minutes) to build networks with a single community and 14 seconds, in average, to build networks with 160 communities.

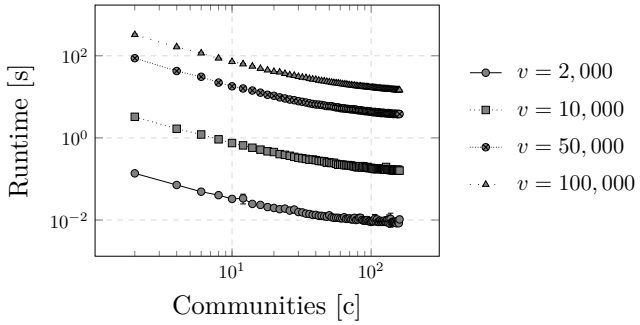


Fig. 11 Runtime as a function of the number of communities c.

We also compared BNOC with the well-known LFR benchmarking tool [28, 27] regarding execution times and scalability. We created networks with parameter v varying in the range $[2,000; 100,000]$ at increments of $2,000$, setting the number of communities $c = 0.01v$ in both tools. Simulating a worst case scenario required generating dense networks, therefore, in BNOC we set dispersion $d = 0.9$ and probability of success $s = 0.9$; in LFR we set a high average degree $k = 40$. The curves depicting the times to create the networks as a function of network size are shown in Figure 12.

While it took, in average, 0.3 and 20 seconds for BNOC to build dense networks with $2,000$ and $100,000$ vertices, respectively, it took 4.4 and 396 seconds (or 6.6 minutes) for LFR to build the equivalent networks. The tools are written in different programming languages (LFR in *C* and BNOC in *Python*) and have different aims (LFR synthesizes simple networks, BNOC synthesizes bipartite networks) but this analysis confirms that BNOC is fast and scalable.

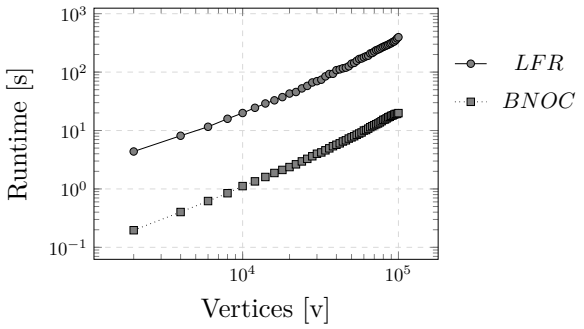


Fig. 12 Runtimes for obtaining synthetic networks of varying sizes in the LFR and BNOC benchmarking tools.

This discussion was aimed at describing and justifying the design choices for BNOC and illustrate its potential for generating diverse large-scale bipartite networks with tens or even hundreds of thousands vertices and hundreds of millions edges in a timely manner.

The time complexity is determined by the cost of generating the binomial distribution and other processes linear in $|V|$. In general, computing a probability of success is constant, therefore, drawing v samples from a negative binomial distribution is $O(|V|)$. However, it depends on the implementation of the exponential and binomial coefficient functions. Our implementation relies on the *numpy* library[9], in which the corresponding functions are linear time. The space complexity of the current implementation is $O(|V|^2)$ as the initial network is a complete network with $|V|$ nodes [15]. Nonetheless, this cost could be reduced employing data structures for representing sparse matrices.

BNOC has been implemented using modular operations, so it can be easily generalized to other scenarios. In Appendix A we outline some generic principles underlying its generalization to creating k-partite and heterogeneous networks, an extension that is already operational in the current version. Other scenarios may be not as straightforward, e.g., generalizing to creating directed networks requires special attention to certain connection properties.

4 Empirical Evaluation

We conducted two experimental studies. The first study investigated whether BNOC can support a standardized, replicable and consistent empirical comparison of the performance (regarding accuracy and runtime behavior) of different algorithms, and also how it can help unveiling the strengths and limitations of community detection algorithms. The second study was conducted for validation purposes, comparing the networks synthesized with BNOC with already established sets of synthetic networks, in two different scenarios.

All experiments (including those reported in Section 3) were conducted in a Linux machine with an 8-core 3.7 GHz CPU and 64 GB main memory. Whenever applicable we report average values obtained from 30 executions for each algorithm.

4.1 Support for consistent algorithm evaluation

We used BNOC to create benchmark networks to assess the performance of two popular algorithms for overlapping community detection in bipartite networks, namely *Hierarchical Link Clustering* (HLC) [1] and *Order Statistics*

[9] http://www.numpy.org/

Local Optimization Method (OSLOM) [29]. HLC focuses on grouping edges rather than vertices to reveal hierarchical overlapped organizations, whereas OSLOM is based on local optimization of a fitness function that expresses the statistical significance of clustering. Both algorithms are applicable to several network domains, however, their performance in bipartite models has not yet been assessed empirically. HLC can be directly applied to bipartite networks, whereas OSLOM can be applied to a one-mode projection of the bipartite network, which we did following the guidelines presented by [36]. Besides comparing the algorithms' capabilities in bipartite networks, it is our goal to verify whether BNOC can create representative benchmarking networks capable of revealing their relevant characteristics, strengths and limitations.

The following three measures of algorithm performance were considered: normalized mutual information (NMI), accuracy (Acc) and runtime (seconds). Acc measures an algorithm's ability to identify both overlapping (sensitivity) and non-overlapping (specificity) vertices. It is defined as $^{(TP+TN)}/_{(TP+FP+TN+FN)}$, where TP is the number of actual overlapping vertices correctly classified, TN is the number of actual non-overlapping vertices correctly classified, FN is the number of actual overlapping vertices mistakenly classified as non-overlapping, and FP is the number of actual non-overlapping vertices mistakenly classified as overlapping. The NMI index relies on information theory to quantify the quality of the communities inferred; we employ the formulation extended by [35] to handle overlapping communities.

NMI considers non-overlapping and overlapping structures, therefore, the number of vertices in each layer can exert direct influence on evaluation results. A set of 270 synthetic weighted bipartite networks was generated with the following parameter settings: $|V|$ varied within the range $[500, 1000]$ at increments of 100, so that $v = [v_1, v_2]$, $|V| = v_1 + v_2$ and $v_1 = 0.8v_2$, which creates unequally sized layers. The remaining parameters were set to n within the range $[0.1, 0.3]$ at increments of 0.025, d varied within the range $[0.5, 0.9]$ at increments of 0.1, $x = y = 0.05|V|$, $z = 0.01|V|$ and $c = 0.02|V|$. Because HLC has high time and space complexities, experiments were limited to small networks.

Figure 13 depicts the NMI values for HLC and OSLOM, respectively, as a function of the amount of noise (parameter n) in networks of different sizes (parameter v).

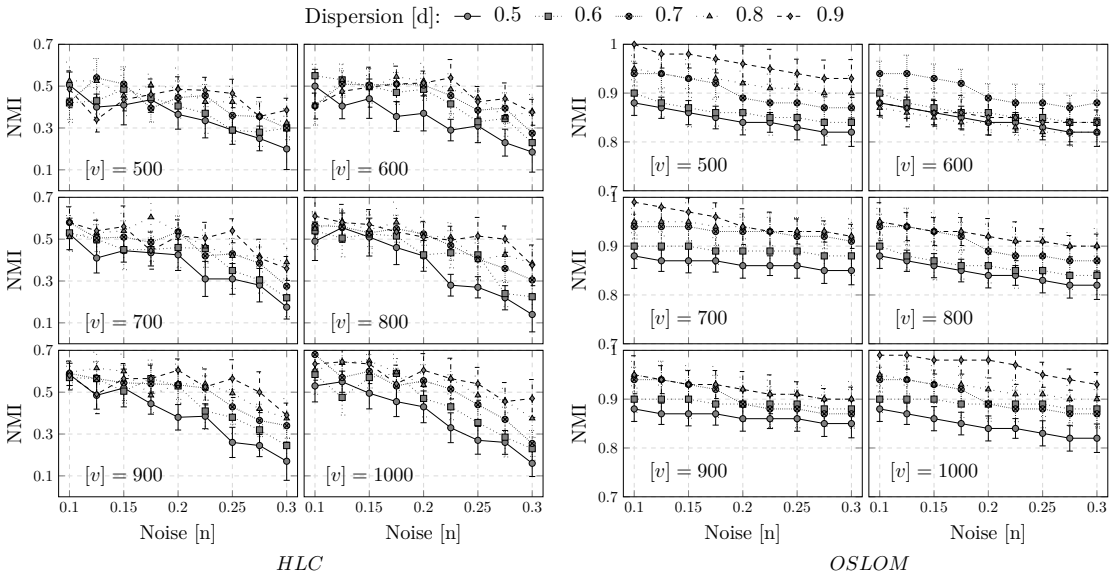


Fig. 13 NMI results for HLC and OSLOM as a function of noise obtained on 270 synthetic bipartite networks of six different sizes $|V|$.

The algorithms exhibit distinct behaviors, i.e., NMI values for HLC decrease quickly as noise levels increase (higher values of parameter n), whereas with OSLOM they decrease slowly. Indeed, increasing noise levels yields more inter-community edges, consequently, boundaries between communities become more blurred and harder to identify. Whereas HLC's performance is evidently highly sensitive to noise levels, OSLOM results are more stable, suggesting its performance is not as strongly affected.

In both cases, higher dispersion values (parameter d, which controls network density) yielded higher values of NMI, as densely connected communities are more cohesive and easily separable. On the other hand, lower dispersion leads to sparser communities with blurred boundaries. In general, both HLC and OSLOM achieve good results regarding the quality of the communities identified, however, their performance is directly affected by the choice of d. In particular, quality degrades when more extensive noise levels are applied to sparser networks (low dispersion values).

The accuracy measure (Acc) quantifies the quality of the overlapping communities inferred by the algorithms, i.e. Acc depends on the number of overlapping vertices or communities, rather than layer size. Therefore, BNOC was employed to create 532 synthetic weighted bipartite networks of sizes $|V| = 600$ and $|V| = 900$, such that $v = [v_1, v_2]$, $|V| = v_1 + v_2$ and $v_1 = v_2$, i.e., equal-sized layers. Parameters $x = y$ were varied within the range $[2, 20]$ at increments of 1, parameter z varied within the range $[2, 15]$ at increments of 1, and remaining parameters were set to $n = 0.1$ and $c = [15, 15]$. Figure 14 shows the Acc values for HLC and OSLOM, respectively, as a function of the numbers of overlapping vertices (parameters x and y) in Figure 14(a)), and overlapping communities (parameter z) in Figure 14(b)).

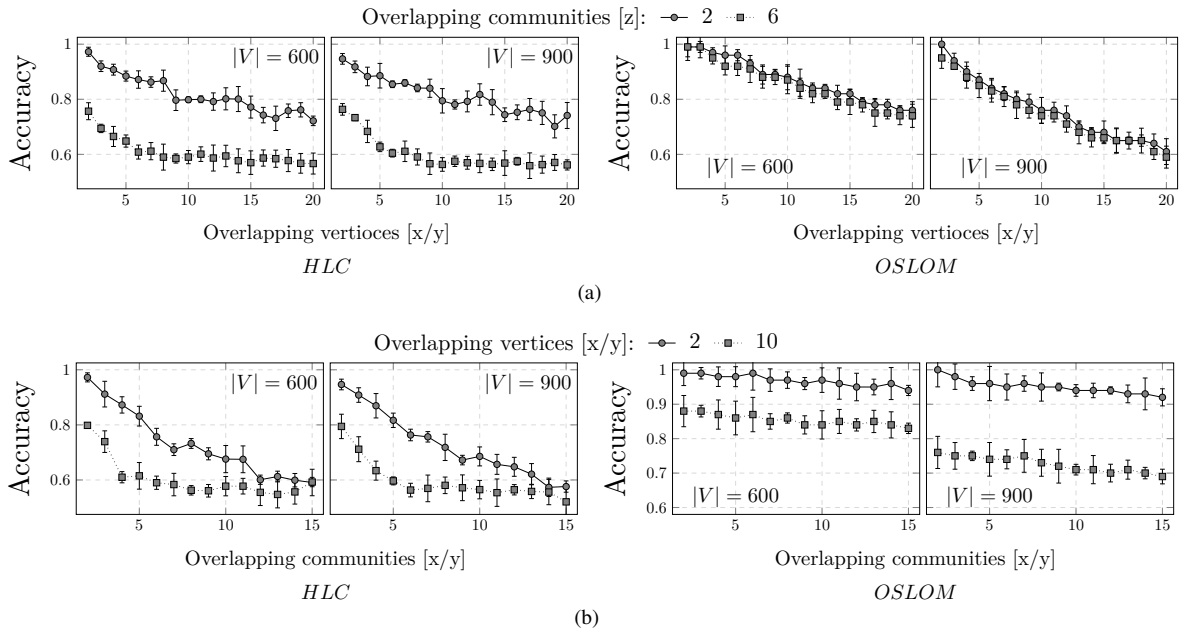


Fig. 14 Acc values for HLC (left) as OSLOM (right) as a function of the numbers of overlapping vertices (a) and overlapping communities (b) in 532 synthetic weighted bipartite networks.

Figure 14(a) (left) shows the Acc values for HLC decrease slowly as the number of overlapping vertices increases. Furthermore, the algorithm exhibits a stable behavior when $x = y \geq 10$, which suggests its performance

is not strongly affected by the number of overlapping vertices. On the other hand, parameter z potentially impacted the quality of the solution, since $z = 2$ (upper curves) yielded higher accuracies than $z = 6$ (bottom curves). The upper and bottom curves differ to up 0.15, although their difference decreases as x increases. This effect is shown in Figure 14(b) (left) and confirms Acc decreases quickly as z increases. Therefore, whilst the influence of parameters x and y on algorithm performance is unclear, it is surely strongly affected by the presence of overlapping communities.

Figure 14(a) (right) shows the Acc values achieved by OSLOM decrease as the number of overlapping vertices increases, which suggests the algorithm's performance is strongly affected by this property. On the other hand, the choice of parameter z, which determines the number of overlapping communities, seems to exert a limited impact on solution quality, since curves for $z = 2$ and for $z = 6$ are nearly overlapping, i.e., yielded similar accuracies. This effect is shown in Figure 14(b) (right) and confirms that Acc values remain stable as z increases.

Results are amenable to statistical analysis, since the algorithms were evaluated in 802 representative and independent networks (270 and 532 networks were used, respectively, to evaluate their performance by means of NMI and Acc). Therefore, a Nemenyi post-hoc test [18] was applied to the results depicted in Figures 13 and 14 to detect statistical differences in the performances of both algorithms. According to the Nemenyi statistics, the critical value for comparing the mean-ranking of two different algorithms at 95 percentile is 0.12. Mean-ranking differences above this value are significant. The average ranks of HLC and OSLOM were $1,0$ and $2,0$, respectively, therefore, OSLOM yielded in statistically higher values than HLC.

The execution times of HLC and OSLOM were measured to analyze the impact of network properties such as size and density on algorithm scalability. The runtime is primarily affected by the total number of vertices, rather than by the layer sizes, therefore, we generated $1,530$ weighted bipartite networks, varying $|V|$ within the range $[100, 1000]$ at increments of 100, so that $v = [v_1, v_2]$, $|V| = v_1 + v_2$ and $v_1 = v_2$ (equal sized layers). Parameter d varied within the range $[0.1, 0.9]$ at increments of 0.05 and parameter n varied within the range $[0.1, 0.3]$ at increments of 0.025. The remaining parameters were set as $x = y = 0.05|V|$, $z = 0.01|V|$ and $c = 0.02|V|$. Figure 15 shows runtime of both HLC (left) and OSLOM (right) as a function of the parameters controlling dispersion, or edge density (parameter d in Figure 15(a)), size (parameter v in Figure 15(b)) and noise (parameter n in Figure 15(c)).

As far as HLC is concerned, Figure 15(a) (left plot) indicates a linear correlation between runtime and dispersion values and Figure 15(b) (left) depicts a nearly quadratic relation with network size. Indeed, an increase in the number of vertices implies an exponential increase in the total number of edges and higher values of the dispersion parameter directly impact on the network edge density. According to Figure 15(c) (left), increasing noise does not affect runtime. Therefore, the combination of network density and network size strongly affects the performance of HLC.

As far as OSLOM is concerned, no clear correlation between runtime and edge density is observed in Figure 15(a) (right plot). Indeed, OSLOM has quadratic complexity in the number of vertices $|V|$, observable in Figure 15(b) (right), whereas an increase in dispersion parameter d directly impacts on the number of edges, without affecting the asymptotic convergence of the algorithm. Furthermore, Figure 15(c) (right) suggests increasing noise levels significantly impacts runtime. We conclude the execution time of OSLOM is strongly impacted by a combination of larger network sizes and high noise levels.

The previous analyses reveal that benchmarking networks obtained with BNOC uncover interesting information on the accuracy and runtime behavior of both algorithms, in relation with network properties such as size, density, noise and overlapping levels. The analyses also enabled a statistical comparison of their performances.

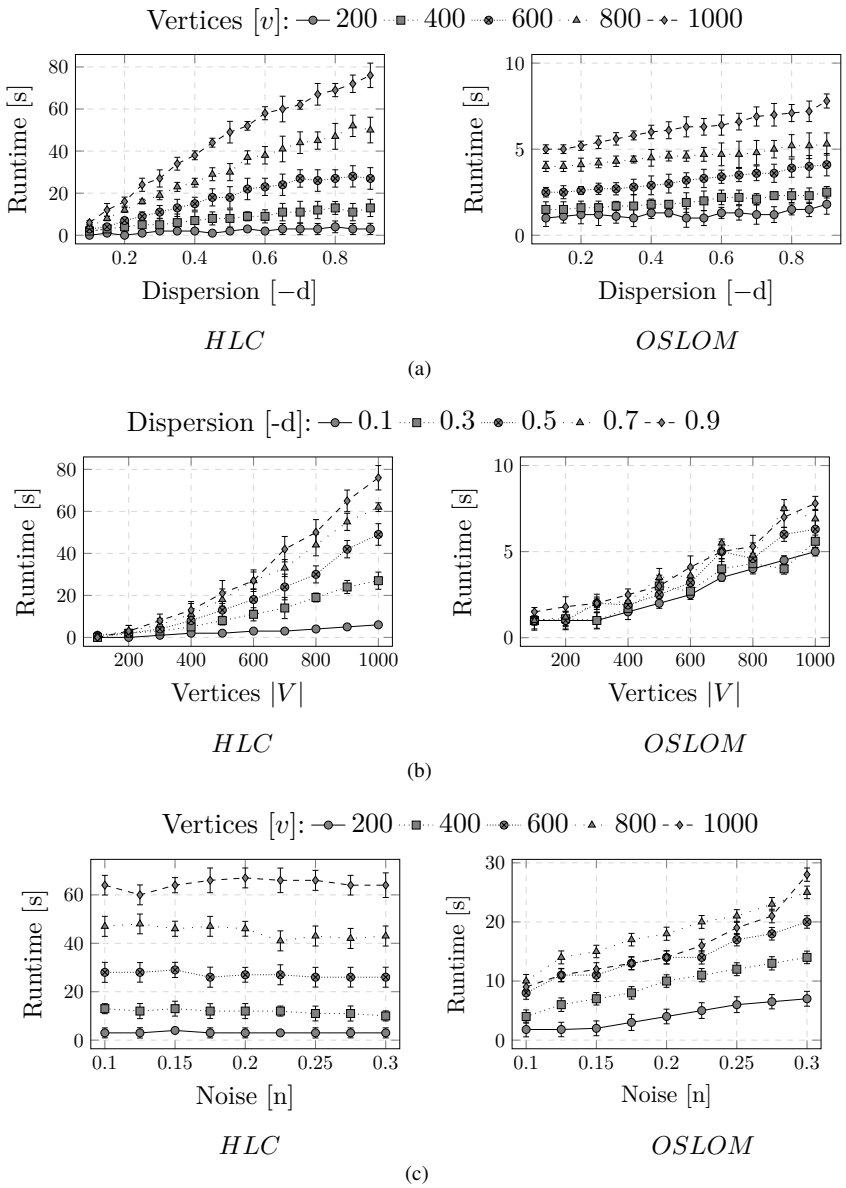


Fig. 15 HLC and OSLOM runtimes, as a function of network density (a), number of vertices (b) and amount of noise (c).

4.2 Comparison with existing benchmarks

We turned off the overlapping community settings in BNOC in order to reproduce the synthetic scenarios considered in previous studies by Melamed [36] and Larremore et al [31]. Our goal is to assess the reliability of the networks

synthesized by BNOC, verifying whether the tool can reproduce networks displaying structural properties similar to those already considered in the literature.

A first study considered the synthetic networks described in Melamed [36], henceforth called Melamed networks. That study manipulated two properties of the networks: (i) the number of communities in each layer: two networks were built with three communities in both layers and two networks with three and two communities in the first and second layer, respectively; (ii) the number of vertices: two networks were built with $v = [60, 120]$ and two networks with $v = [600, 1200]$.

The networks have been synthesized in BNOC using the following parameter settings: (1) $v = [60, 120]$, $c = [3, 3]$, $n = 0.2$, $d = 0.6$, $p = 0.6$; (2) $v = [600, 1200]$, $c = [3, 3]$, $n = 0.1$, $d = 0.4$, $p = 0.4$; (3) $v = [60, 120]$, $c = [3, 2]$, $n = 0.5$, $d = 0.5$, $p = 0.5$; and (4) $v = [600, 1200]$, $c = [3, 2]$, $n = 0.3$, $d = 0.3$, $p = 0.3$.

Figures 16(a)-16(d) illustrate the four Melamed networks; Figures 16(e)-16(h) illustrate the corresponding BNOC counterparts. The network visualizations allow a global observation of the topological similarities between the pairs of synthetic networks thus obtained. The BNOC networks closely resemble the Melamed networks, particularly the pairs constructed with the same number of communities in both layers, namely (a)-(e) and (b)-(f). Layouts in Figure 16 have been computed with the Fruchterman-Reingold force-directed algorithm under same parameter settings [22].

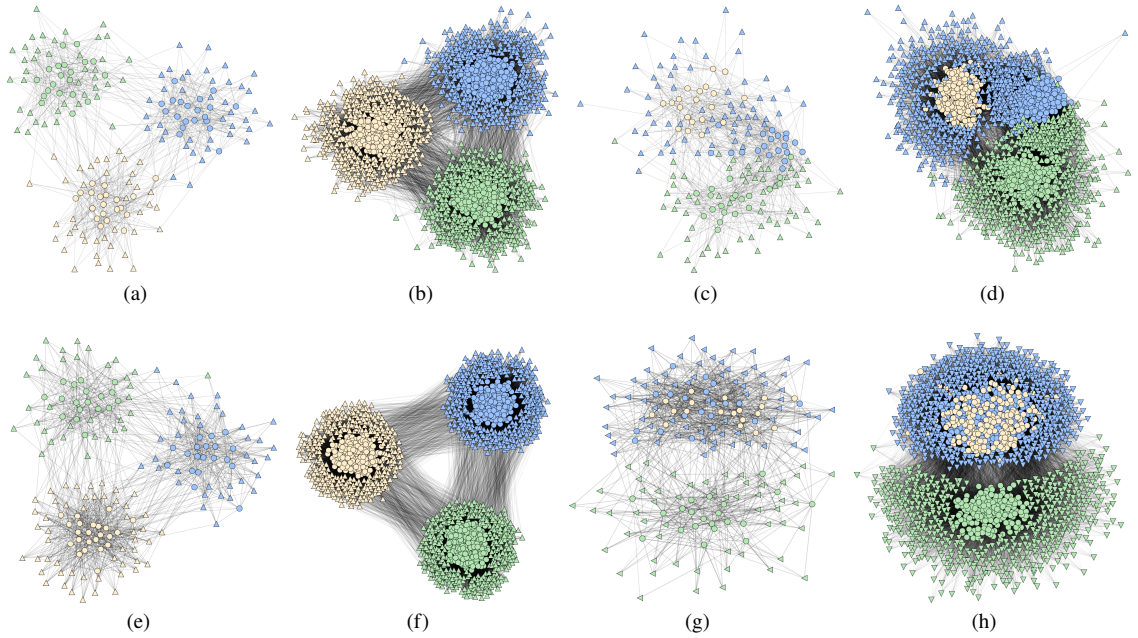


Fig. 16 Two equivalent sets of synthetic networks obtained with different benchmarking tools. (a), (b), (c), and (d) display the set of Melamed networks; and (e), (f), (g), and (h) display the corresponding networks synthesized using BNOC.

In order to obtain an objective estimate of their topological equivalence we evaluated the well-known community detection algorithm biSBM[10] [31] on both sets of networks. Table 2 reports the mean normalized mutual information

[10] https://github.com/junipertcy/det_k_bisbm

(NMI) [17] measures and corresponding standard deviation errors obtained with 10 network replications and 10 biSBM executions.The results obtained are highly similar, providing strong evidence that the network pairs in Figure 16 – (a)-(e), (b)-(f), (c)-(g) and (d)-(h) – have equivalent topological properties.

Table 2 Normalized mutual information (NMI) mean values and standard deviations resulting from running biSBM in the BNOC and the Melamed network sets.

Benckmark	Network A	Network B	Network c	Network D
BNOC Networks	0.80 ± 0.03	0.97 ± 0.01	0.85 ± 0.03	0.98 ± 0.01
Melamed Networks	0.79 ± 0.02	0.98 ± 0.02	0.85 ± 0.02	0.98 ± 0.01

A similar analysis was conducted in the synthetic networks provided in Larremore et al [31], henceforth called Larremore networks. That study considered two scenarios: The first typifies an easy case for community detection, of networks with $v = [1000, 1000]$ and four equal sized communities. The second represents a more challenging scenario, where the networks were built with $v = [300, 700]$ and vertices split into $\{100, 150, 50\}$ and $\{350, 350\}$ communities in the first and the second layers, respectively. Each scenario is represented by 20 networks obtained varying the mixing parameter γ within the range $[0, 1]$ at increments of 0.05, wherein $\gamma = 0$ denotes the maximum setting for noise and $\gamma = 1$ denotes the absence of inter-community edges.

The first set of networks was replicated employing the following parameter settings in BNOC: $d = 0.25$, $v = [1000, 1000]$, $c = [4, 4]$, $p^0 = p^1 = [0.25, 0.25, 0.25, 0.25]$; for the second one the settings were: $v = [300, 700]$, $c = [3, 2]$, $p^0 = [0.4, 0.3, 0.2]$ and $p^1 = [0.5, 0.5]$. Again, for each case 20 networks were created with noise level parameter n varying in the range $[0, 1]$ at increments of 0.05.

There are two major differences in the processes used to synthesize the Larremore and the BNOC networks. First, the generating model in Larremore et al [31] creates the communities by adding an exact number of vertices to each community, whereas BNOC considers the probabilities of vertices belonging to each community. Second, the mixing parameter γ, used to control noise in [31], is the opposite of how noise is defined in BNOC.

Figure 17 reports the corresponding biSBM NMI mean values and standard deviations (y-axis) obtained, as a function of γ (upper x-axis) or n (lower x-axis) for both network sets, with (a) depicting the easy scenario and (b) depicting the difficult one.

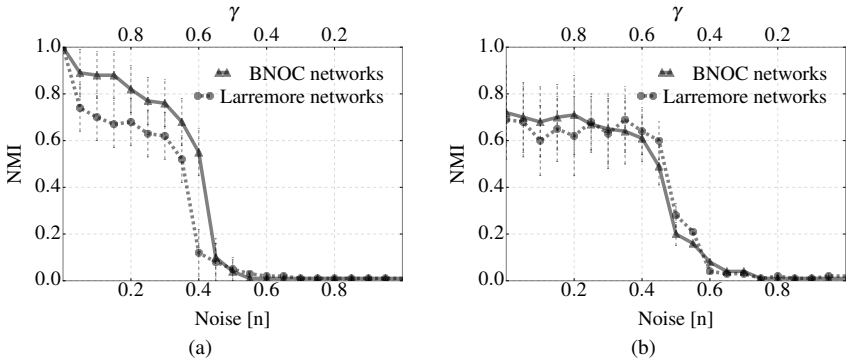


Fig. 17 NMI values for biSBM as a function of mixing parameter γ (lower x-axis) or noise level parameter n (upper x-axis) obtained on the Larremore and the BNOC networks. (a) shows results for the easy scenario; (b) shows results for the difficult scenario. Each scenario is described by 20 networks (varying noise levels) and mean values and errors are relative to 10 network replications and 10 biSBM executions.

Similarly to the previous study, similar NMI behavior is observed in the synthetic network sets, i.e., the curves in both the easy and the difficult scenarios display equivalent behavior. In summary, for very low noise levels (indicated by n or γ) they decrease slowly as noise increases; then the curves drop sharply as the noise levels increase further and the number of inter-community edges approaches 50%; finally NMI stabilizes at very low values on the noisier networks.

The previous analyses confirm BNOC as a reliable general-purpose benchmarking tool useful to replicate network models already considered in the literature or to synthesize new models with arbitrary properties.

5 Conclusion

In this paper we introduced BNOC, a tool to generate synthetic weighted bipartite networks with overlapping structures for benchmarking purposes.

Its potential was illustrated with an assessment of two well-known overlapping community detection algorithms, HLC and OSLOM, on benchmark networks with very distinctive properties. Our results indicate the performance of HLC is highly sensitive to both noise and number of overlapping vertices and communities. Furthermore, its execution times are strongly affected by network size and community density. On the other hand, the performance of OSLOM is highly sensitive to edge density and number of overlapping vertices and its execution times are strongly affected by both network size and noise levels. Such findings deserve attention in further studies and application of these algorithms.

This study also illustrates BNOC's potential usefulness as a flexible, robust and reliable resource for creating arbitrary bipartite networks for benchmarking purposes. It can synthesize large-scale bipartite networks with tens or even hundreds of thousands vertices and tens of millions edges in very reasonable times. We observed, however, that scalability may be impaired when creating networks that lack a community structure or have very few communities.

We also compared BNOC with existing benchmarking models. The results of comparing the synthetic models obtained from different tools provide strong evidence on the reliability of BNOC as a general-purpose benchmarking tool for creating networks with arbitrary topological properties.

BNOC may be useful for benchmarking other kinds of algorithms and applications beyond community detection. For instance, if the layers represent, respectively, observations and features, and communities are interpreted as classes, it may support comparative assessment of classification algorithms. It may support the design very large networks with well-known target properties for purposes of testing e.g., graph drawing or link prediction algorithms, since topological features such as community structure and membership can be imposed *a priori* to reflect desired characteristics.

As future work we intend to extend the tool to handle directed bipartite networks. Additional issues deserve further investigation, including: conducting systematic studies on how to tune the parameter values to yield networks exhibiting a particular combination of target features; in-depth benchmarking of algorithm behavior in networks with highly unbalanced layers; an analysis of edge weights on distinct ranges in setting the corresponding probability parameters s and d. The current implementation of BNOC can be downloaded at https://github.com/alanvalejo/bnoc[11]

Acknowledgements This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. This work has been partially supported by the State of São Paulo Research Foundation (FAPESP) grants 15/14228-9 and 17/05838-3; and the Brazilian Federal Research Council (CNPq) grants 302645/2015-2 and 305696/2013-0.

[11] The tool will be made available immediately after paper acceptance.

References

1. Ahn YY, Bagrow JP, Lehmann S (2010) Link communities reveal multiscale complexity in networks. Nature 466(7307):761–764
2. Akoglu L (2014) Quantifying political polarity based on bipartite opinion networks. In: Proceedings of the International AAAI Conference on Web and Social Media (AAAI), Eighth International AAAI Conference on Weblogs and Social Media (ICWSM)
3. Akoglu L, Faloutsos C (2009) Rtg: A recursive realistic graph generator using random typing. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5781 LNAI(PART 1):13–28
4. Alessandro M, Vittorio CC (2018) Leveraging the nonuniform PSO network model as a benchmark for performance evaluation in community detection and link prediction. New Journal of Physics 20(6):063,022
5. Ali AM, Alvari H, Hajibagheri A, Lakkaraj K, Sukthankar G (2014) Synthetic generators for cloning social network data. In: Proceedings of the International Conference on Social Informatics (Socinfo)
6. Armstrong TG, Ponnekanti V, Borthakur D, Callaghan M (2013) Linkbench : a database benchmark based on the facebook social graph. In: Proceedings of the International Conference on Management of Data (SIGMOD), pp 1185 – 1196
7. Barabasi AL, Bonabeau E (2003) Scale-free netwroks. Scientific American (SCIAM)
8. Barber MJ (2007) Modularity and community detection in bipartite networks. Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 76(6):1–11
9. Barrett CL, Beckman RJ, Khan M, Kumar VSA, Marathe MV, Stretz PE, Dutta T, Lewis B (2009) Generation and analysis of large synthetic social contact networks. In: Proceedings of the Winter Simulation Conference, WSC '09, pp 1003–1014
10. Beckett SJ (2016) Improved community detection in weighted bipartite networks. Royal society open science 3(1):140,536
11. Birmelé E (2009) A scale-free graph model based on bipartite graphs. Discrete Applied Mathematics 157(10):2267 – 2284
12. Boncz P (2013) Ldbc: Benchmarks for graph and rdf data management. In: Proceedings of the International Database Engineering & Applications Symposium, pp 1–2
13. Capota M, Hegeman T, Iosup A, Prat-Pérez A, Erling O, Boncz P (2015) Graphalytics: A big data benchmark for graph-processing platforms. In: Proceedings of the Graph Data management Experiences and Systems (GRADES), pp 1–6
14. Chakrabarti D, Zhan Y, Faloutsos C (2004) R-mat: A recursive model for graph mining. In: Proceedings of the Society for Industrial and Applied Mathematics (SIAM), International Conference on Data Mining (SDM), p 5
15. Cormen TH, Leiserson CE, Rivest RL, Stein C (2009) Introduction to Algorithms, Third Edition, 3rd edn. The MIT Press
16. Cui Y, Wang X (2014) Uncovering overlapping community structures by the key bi-community and intimate degree in bipartite networks. Physica A: Statistical Mechanics and its Applications 407(0):7–14
17. Danon L, Díaz-Guilera A, Duch J, Arenas A (2005) Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment 2005:P09,008
18. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research 7:1–30
19. Du N, Wang B, Wu B, Wang Y (2008) Overlapping community detection in bipartite networks. Proceedings of the International Conference on Web Intelligence (IEEE/WIC/ACM) (60402011):176–179

20. Faleiros TP, Rossi RG, de Andrade Lopes A (2017) Optimizing the class information divergence for transductive classification of texts using propagation in bipartite graphs. Pattern Recognition Letters 87(Supplement C):127 – 138, advances in Graph-based Pattern Recognition
21. Fortunato S (2010) Community detection in graphs. Physics Reports 486(3-5):75–174
22. Fruchterman TMJ, Reingold EM (1991) Graph drawing by force-directed placement. Softw Pract Exper 21(11):1129–1164
23. Girvan M, Newman MEJ (2002) Community structure in social and biological networks. In: Proceedings of the National Academy of Science of the United States of America, vol 99, pp 7821–7826
24. Grujić J (2008) Movies recommendation networks as bipartite graphs. In: Proceedings of the International Conference on Computational Science (ICCS), Springer Berlin Heidelberg, pp 576–583
25. Hwang T, Sicotte H, Tian Z, Wu B, Kocher JP, Wigle DA, Kumar V, Kuang R (2008) Robust and efficient identification of biomarkers by classifying features on graphs. Bioinformatics 24(18):2023–2029
26. Jonnalagadda A, Kuppusamy L (2016) A survey on game theoretic models for community detection in social networks. Social Network Analysis and Mining 6(1):83
27. Lancichinetti A, Fortunato S (2009) Community detection algorithms: A comparative analysis. Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 80(5):1–11
28. Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 78(4):1–5
29. Lancichinetti A, Radicchi F, Ramasco JJ, Fortunato S (2011) Finding statistically significant communities in networks. Plos One 6(4):1–18
30. Largeron C, Mougel PN, Rabbany R, Zaïane OR (2015) Generating attributed networks with communities. Plos One 10(4):1–21
31. Larremore DB, Clauset A, Jacobs AZ (2014) Efficiently inferring community structure in bipartite networks. Phys Rev E 90:012,805
32. Latapy M, Magnien C, Vecchio ND (2008) Basic notions for the analysis of large two-mode networks. Social Networks 30(1):31–48
33. Lehmann S, Schwartz M, Hansen LK (2008) Biclique communities. Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 78(1):1–9
34. Li Z, Zhang S, Zhang X (2015) Mathematical model and algorithm for link community detection in bipartite networks. American Journal of Operations Research 5:421–434
35. McDaid AF, Greene D, Hurley N (2011) Normalized mutual information to evaluate overlapping community finding algorithms. In: eprint arXiv:1110.2515
36. Melamed D (2014) Community structures in bipartite networks: A dual-projection approach. Plos One 9(5):1–5
37. Moussiades L, Vakali A (2009) Benchmark graphs for the evaluation of clustering algorithms. In: Proceedings of the International Conference on Research Challenges in Information Science (RCIS), pp 197–206
38. Nettleton DF (2016) A synthetic data generator for online social network graphs. Social Network Analysis and Mining 6(1):44
39. Newman MEJ (2001) Scientific collaboration networks. i. network construction and fundamental results. Physical Review E 64:016,131
40. Newman MEJ (2001) Scientific collaboration networks. ii. shortest paths, weighted networks, and centrality. Physical Review E 64:016,132
41. Newman MEJ (2010) Networks: An Introduction. Oxford University Press, Inc., New York, NY, USA
42. Pasta MQ, Zaidi F (2016) Leveraging evolution dynamics to generate benchmark complex networks with community structures. In: eprint arXiv:1606.01169, vol abs/1606.01169
43. Pérez-Rosés H, Sebé F (2014) Synthetic generation of social network data with endorsements. In: eprint arXiv:1411.6273

44. Pham MD, Boncz P, Erling O (2013) S3g2: A scalable structure-correlated social graph generator. Proceedings in Selected Topics in Performance Evaluation and Benchmarking: 4th TPC Technology Conference (August):222
45. Rabbany R, Takaffoli M, Fagnan J, Zaïane OR, Campello RJGB (2013) Communities validity: methodical evaluation of community mining algorithms. Social Network Analysis and Mining 3(4):1039–1062
46. Rees BS, Gallagher KB (2012) Overlapping community detection using a community optimized graph swarm. Social Network Analysis and Mining 2(4):405–417
47. Rosvall M, Delvenne JC, Schaub MT, Lambiotte R (2017) Different approaches to community detection. arXiv e-prints p arXiv:1712.06468, 1712.06468
48. Shi C, Li Y, Zhang J, Sun Y, Philip SY (2017) A survey of heterogeneous information network analysis. IEEE Transactions on Knowledge and Data Engineering 29(1):17–37
49. Souam F, Aitelhadj A, Baba-Ali R (2014) Dual modularity optimization for detecting overlapping communities in bipartite networks. Knowledge and Information Systems 40(2):455–488
50. Uslu T, Mehler A (2018) PolyViz: a visualization system for a special kind of multipartite graphs. In: Proceedings of the IEEE VIS 2018, accepted
51. Valejo A, Drury B, Valverde-Rebaza J, de Alneu de Andrade Lopes (2014) Identification of related brazilian portuguese verb groups using overlapping community detection. In: Proceeding of the International Conference on Computational Processing of the Portuguese Language, Springer, Cham, pp 292–297
52. Valejo A, Valverde-Rebaza JC, de Andrade Lopes A (2014) A multilevel approach for overlapping community detection. In: Proceedings of the Brazilian Conference on Intelligent Systems (BRACIS), Springer, Berlin
53. Valejo A, Oliveira MCRF, Filho GP, Lopes AA (2018) Multilevel approach for combinatorial optimization in bipartite network. Knowledge-based systems 151:45–61, DOI https://doi.org/10.1016/j.knosys.2018.03.021
54. Yang Z, Perotti JI, Tessone CJ (2017) Hierarchical benchmark graphs for testing community detection algorithms. Physical Review E 96:052,311
55. Zhang ZY, Ahn YY (2015) Community detection in bipartite networks using weighted symmetric binary matrix factorization. International Journal of Modern Physics C 26:1–14
56. Zhong E, Fan W, Zhu Y, Yang Q (2013) Modeling the dynamics of composite social networks. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 937–945

Appendix A HNOC: Extension to k-partite and heterogeneous networks

As in the case of bipartite networks there is a lack of benchmarking tools to create k-partite and heterogeneous networks for assessing community detection and other algorithms. k-partite networks have k vertex types, rather than just two, with edges connecting only vertices of different types. In heterogeneous networks this second restriction is dropped, i.e., edges can occur between vertices of the same type. Since they are direct generalizations of bipartite networks, we extended BNOC to synthesize general heterogeneous networks. This extension, called HNOC, can be useful to generate HIN models to support development and validation of new methods. HNOC inherits BNOC's major features as a flexible and robust resource to synthesize a variety of benchmarking networks with distinct properties in reasonable times.

A heterogeneous information network (HIN) [48] consists of m disjoint sub-sets of vertices of different types (called layers) and edges connecting these elements. A network $\mathcal{X}$ with m vertex types can be partitioned into sub-sets $X_i = \{x_{i,1}, \ldots, x_{i,n_i}\}$ for each type i. A HIN is represented as a graph $G = (V, E, W)$, where $V = \bigcup_{i=1}^{m} X_i$ with $m > 2$, E is the set of edges, and W is the set of edge weights.

HINs have a inherently complex structure that can be difficult to handle and visualize. The structure of connections can be described by a network schema [48], which defines a meta template for the network that describes its vertex and connection types. Given a graph G, its network schema, denoted $T_G(\mathcal{A}, \mathcal{R})$, is a directed graph defined over element types $\mathcal{A}$ with edges as relations from $\mathcal{R}$, obtained via mapping functions $\varphi : V \rightarrow \mathcal{A}$ and $\psi : E \rightarrow \mathcal{R}$, respectively. Figure 18 exemplifies network schemas for a bipartite network, a k-partite network and an heterogeneous network.

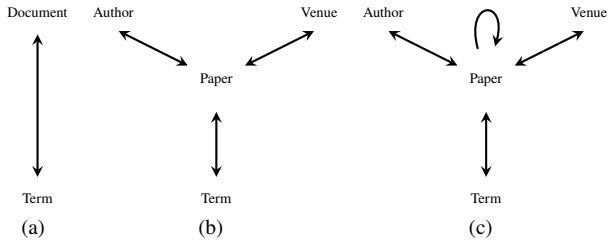


Fig. 18 Network schemas: (a) bipartite network schema; (b) k-partite network schema; and (c) heterogeneous network schema.

The schema explicitly identifies the m vertex types and the r connection types. Each connection type is described by the types of the two endpoints and the connection meaning, since pairs of entities can admit multiple types of connections, as in the case of multi-relational networks [56]. Thus, a HIN can be interpreted as a composition of r bipartite (or homogeneous, if both endpoints of the connections are of the same type) networks, as illustrated in Figure 19.

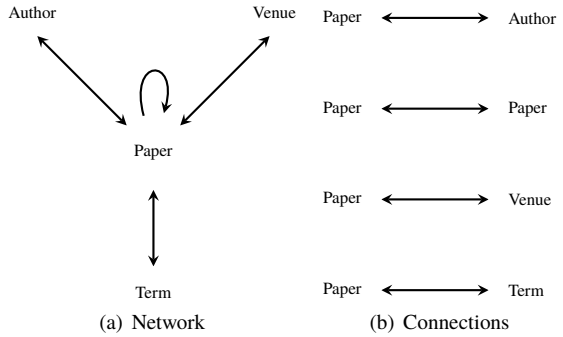


Fig. 19 A HIN described as r bipartite/homogeneous networks.

Following this interpretation we extended BNOC to generate synthetic heterogeneous networks. The extension iterates over each pair of vertex and connection types specified in the network schema, employing similar steps to build the communities in each iteration:

1: Execute m iterations of BNOC's Steps 1 and 2 to build each layer i with V_i vertices, set a single community on each layer and introduce the overlapping structures.

2: For each pair of layers specified in the schema, execute BNOC's Steps 3, 4 and 5 in order to establish the specified connections, weights, density and noise levels.

Since heterogeneous networks have multiple layers and multiple connection types, the extension required modifying some BNOC parameters and introducing a few additional parameters, as described in Table 3. The following parameters were added: the number of layers m and the set connected layers, henceforth called "*schema*", e. Furthermore, the dispersion and noise parameters must be defined for each schema, since each iteration handles a pair of connected layers.

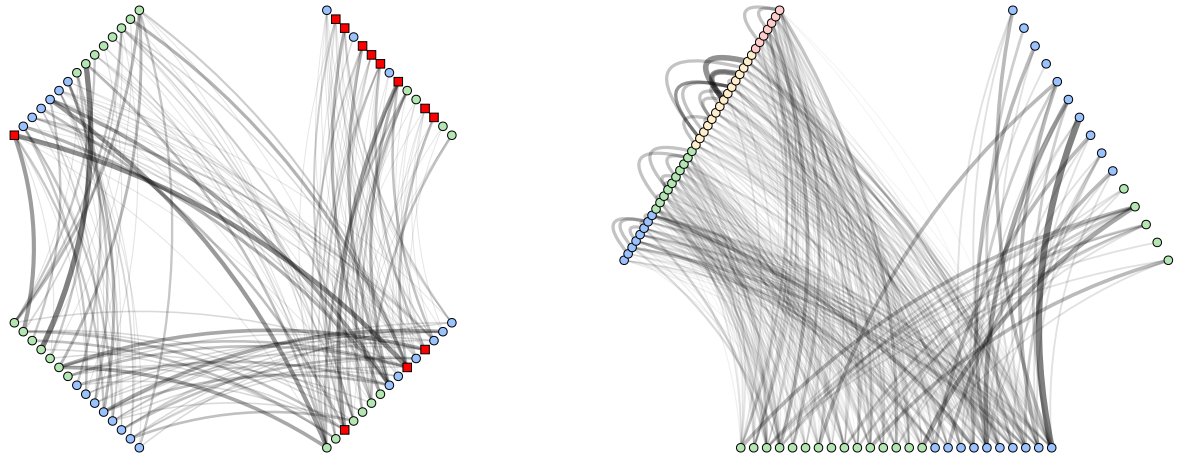
Table 3 Modified and additional parameters in HNOC.

Parameter	Type	Domain	Default	Description
-m, –layers	number	$\mathbb{N}$	3	Number of vertex types
-v, –vertices	$\|p\| = m$	$(0, \|V\|] \in \mathbb{N}$	[10, 10, 10]	Number of vertices for each layer
-e, –schema	$\|p\| = r \times 2$	$\mathbb{N}$	*null*	List of pairs that defined the connected layers
-p, –probabilities	$\|p\| = m \times c$	$(0, \|V\|] \in \mathbb{R}$	*null*	Probabilities for vertices in each layer for each community
-d, --dispersion	$\|p\| = m$	$\mathbb{R}_+$	0.1	Dispersion of negative binomial distribution for each scheme
-n, --noise	$\|p\| = m$	$(0, 1] \subseteq \mathbb{R}$	0.01	Noise for each scheme
-b, –balanced	boolean, $\|p\| = m$	0, 1	[0, 0, 0]	Boolean balancing flag that suppresses parameter -p
-x, –overlap	$\|p\| = m$	$(0, \|V\|] \in \mathbb{N}$	[0, 0, 0]	Number of overlapping vertices in each layer
-z, –noverlap	$\|p\| = m$	$(0, \|c\|] \in \mathbb{N}$	[2, 2, 2]	Overlapping communities in each layer

Figure 20 illustrates two networks created with distinct parameter combinations. Unless informed otherwise the parameter settings correspond to the default values informed in Tables 1 and 3. Figure 20(a) depicts a 4-partite network with some community overlapping.

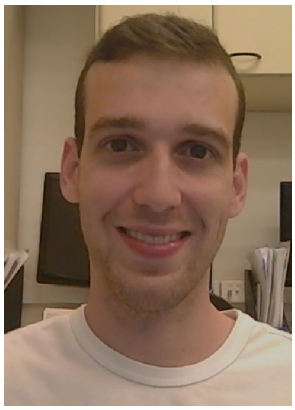
The network was built so that all layers have the same number of communities, the probability set to produce balanced communities, different numbers of overlapping vertices in each layer, and the same dispersion (edge density) d in all pairs of connected layers. Figure 20(b) illustrates a 3-partite network with heterogeneous structure and edges between vertices of the same type in one of the layers. The example can describe a hypothetical author-paper-term network, in which authors are connected with their papers, terms are connected with their neighbouring terms in the text and with the papers in which they appear. The upper left, central and upper right layers, represent, respectively, the Term, Paper and Author entities. The network has been created so that the connections between the different pairs of entities display different patterns, e.g., there is a dense topology of connections between terms and papers, whereas connections between terms and terms, or between authors and papers are sparser. The schema is inspired in the largely used real-world data of DBLP[12] (Digital Bibliography & Library Project), a computer science bibliographic dataset that relates documents, authors and terms.

[12] https://dblp.uni-trier.de/



(a) (b)

Fig. 20 Heterogeneous networks generated with HNOC presenting distinct topological structures and properties: red squares depict overlapping vertices and colored circles indicate non-overlapping vertices and their assigned community; line widths reflect the corresponding edge weights. (a) illustrates a 4-partite network with $v = [15, 15, 15, 15]$, $e = [(0, 1), (1, 2), (2, 3), (3, 1)]$ and $x = [8, 3, 0, 1]$; (b) depicts a heterogeneous network obtained with settings $v = [40, 25, 15]$, $e = [(0, 1), (1, 2), (2, 2)]$, $c = [2, 2, 4]$, and $d = [0.45, 0.85, 0.15, 0.15]$. The network drawings were obtained based on the technique described by [50].



Alan Valejo received the BSc, M.S. and Ph.D. degrees in computer science from University of São Paulo, Brazil, in 2011, 2014 and 2019, respectively. His current research is focused on multilevel methods applied to problems in data science, complex networks analysis, machine learning, and data mining.

Fabiana Góes received the BSc and M.S. degrees in computer science from the Federal University of Pará, Brazil, in 2013 and 2016, respectively. She is currently working toward the Ph.D. degree at University of São Paulo, Brazil. Her research interests include Machine Learning, Graph-based Learning, Heterogeneous Networks with applications in biomedical data and recommendation systems.

Luzia Romanetto received the BSc in Applied Mathematics and Scientific Computing, and the M.S. degree in Computer Science from University of São Paulo, Brazil, in 2010 and 2013, respectively. She is currently working toward the Ph.D. degree at University of São Paulo, conducting research mainly focused on Machine Learning, Heterogeneous Networks, Data Mining and Information Visualization.

Maria Cristina Ferreira de Oliveira received the BSc in Computer Science from the University of São Paulo, Brazil, in 1985, and the Ph.D. degree in Electronic Engineering from the University of Wales, Bangor, in 1990. Since 2008 she is a Full Professor at the Computer Science Department of the Instituto de Ciências Matemáticas e de Computação, at the University of São Paulo, Brazil. She is a member of the ACM, IEEE and of the Brazilian Computer Society. Her current research interests are in Visual Analytics, Visual Data Mining and Information Visualization with applications in audio, text, image and network datasets.

Alneu de Andrade Lopes holds a degree in Civil Engineering from the Federal University of Mato Grosso do Sul (1985), Brazil, a master's degree in Computer Science and Computational Mathematics from University of São Paulo (1995) and a PhD degree in Computer Science from University of Porto (2001), Portugal. Since 2002 he has been with the faculty of Computer Science at the University of São Paulo. His research is in the field of Artificial Intelligence and Machine Learning, mainly in the following subjects: Data Mining and Complex Network Mining.