Data-Driven Intra-Autonomous Systems Graph Generator

IEEE Transactions on Network and Service Management

Vinicius Dadauto, Caio; da Fonseca, Nelson L.S.; da Silva Torres, Ricardo

https://doi.org/10.1109/TNSM.2024.3425508

Data-Driven Intra-Autonomous Systems Graph Generator

Caio Vinicius Dadauto, Nelson L. S. da Fonseca, *Senior Member, IEEE*, and Ricardo da S. Torres, *Member, IEEE*

***Abstract*—Accurate modeling of realistic network topologies is essential for evaluating novel Internet solutions. Numerous investigations have used topologies generated by graph generators employing scale-free-based models. Although scale-free networks accurately encode node degree distribution, they overlook crucial graph properties, such as betweenness, clustering, and assortativity. The limitations of existing generators pose challenges for evaluating network mechanisms and protocols, such as routing. This paper introduces a novel deep learning-based generator of synthetic graphs representing intra-autonomous on the Internet, named Deep-Generative Graphs for the Internet (DGGI). It also presents a massive new dataset of real intra-AS graphs extracted from the project Internet Topology Data Kit (ITDK), called Internet Graphs (IGraphs). DGGI creates synthetic graphs that accurately reproduce the properties of centrality, clustering, assortativity, and node degree. DGGI overperforms existing Internet topology generators. On average, DGGI improves the Maximum Mean Discrepancy (MMD) metric by 84.4%, 95.1%, 97.9%, and 94.7% for assortativity, betweenness, clustering, and node degree, respectively.**

***Index Terms*—Machine learning, graphs and networks, Internet topology, topology generator.**

I. Introduction

GENERATING graphs that represent realistic network topologies is crucial for modeling and assessing novel protocols and traffic control mechanisms for the Internet [1], [2]. The growing employment of deep learning models, particularly Graph Neural Networks (GNN), in various networks and communication mechanisms and protocols requires extensive and diverse datasets [3], [4], [5]. The availability of such datasets achieved by GNN can benefit investigations of novel traffic control mechanisms and protocol proposals [3], [4], [5]. However, topology generators that encode multiple properties of the Internet topology are still unavailable [6], [7], [8], [9], [10].

The node connections in the Internet topology graphs are largely modeled by a heavily tailed distribution [11]. The algorithm Barábasi-Abert (BA) [12] algorithm is the most popular one for generating scale-free networks (*i.e.*, networks for which power-law distribution can model node relations), and has served as the basis for several topology generators [13], [14], [15], [16].

Most Internet topology graph generators are structure-based and focus on inter-AS topologies [7], [8], [9], [10], [17], [18]. Moreover, empirical observations are typically used to model the structure of the Internet as a hierarchical composition of graphs. This hierarchy is based on various assumptions, such as scale-free, uniform growth, and function fitting for the node degree distribution. However, these assumptions only partially capture the characteristics of subgraphs in the Internet topology since they rely on a network growth pattern ultimately based on a power-law distribution. Structure-based generators have been designed to synthesize graphs of hundreds of thousands of nodes but do not accurately reproduce the properties of intra-AS graphs, such as betweenness, node degrees, clustering, and assortativity (Section VI).

Classical generators based on scale-free models exhibit commendable characteristics, such as elegant mathematical formulations and a high degree of flexibility. Their proven utility and efficiency in various studies underscore the strength of their well-established theoretical foundations [19], [20], [21].

While graph generators based on power-law assumptions effectively model node degree distribution, they often overlook other important graph properties [22], such as betweenness, clustering, and assortativity. The replication of these metrics is essential for addressing communication network issues. Centrality metrics like betweenness, for instance, are valuable in finding alternative routes when a node failure occurs [23], [24], while clustering metrics have been utilized in routing protocol solutions [25]. Furthermore, assessing network robustness may involve clustering coefficients, which help to gauge network resilience against security threats [26]. Assortativity also plays a critical role in quantifying network growth, underscoring the importance of accurately replicating it in any graph generator [7], [9], [10].

However, data-driven solutions for studies on Internet protocols and traffic control mechanisms have relied on trivial topologies. They are typically based on only a few samples

Manuscript received 23 July 2023; revised 22 February 2024 and 27 June 2024; accepted 30 June 2024. Date of publication 9 July 2024; date of current version 16 October 2024. The authors would like to thank CAPES (grant #88882.329131/2019–01). Authors are also grateful to CNPq (grant #307560/2016-3), (CNPq INCT grant #405940/2022-0), São Paulo Research Foundation – FAPESP (grants #2014/12236-1, #2015/24494-8, and #2016/50250-1, #2017/20945-0, #2023/00673-7). The associate editor coordinating the review of this article and approving it for publication was Y. Diao. *(Corresponding author: Nelson L. S. da Fonseca.)*

Caio Vinicius Dadauto and Nelson L. S. da Fonseca are with the Institute of Computing, University of Campinas, Campinas 13083-852, Brazil (e-mail: caio.dadauto@ic.unicamp.br; nfonseca@ic.unicamp.br).

Ricardo da S. Torres is with the Department of ICT and Natural Sciences, Norwegian University of Science and Technologies, 6009 Ålesund, Norway, and also with the Artificial Intelligence Group, Wageningen University and Research, 6708 PB Wageningen, The Netherlands (e-mail: ricardo.torres@ntnu.no; ricardo.dasilvatorres@wur.nl).

Digital Object Identifier 10.1109/TNSM.2024.3425508

of real networks or synthetic ones generated by BA-based models [3], [5], [27]. The use of unrealistic topologies can produce misleading assessments of the effectiveness of the performance of new solutions [26], [28], [29].

This paper proposes an intra-AS graph generator based on deep learning, named Deep-Generative Graphs for the Internet (DGGI). It accurately reproduces Internet graphs' centrality, clustering, assortativity, and node degree. DGGI allows the customization of synthetic graphs and can generate an arbitrary number of synthetic parameterized graphs. To our knowledge, this is the first paper to propose an intra-AS graph generator based on deep learning.

This paper also introduces a novel dataset of 90,326 intra-AS subgraphs extracted from the sets of large intra-AS graphs (millions of nodes) named IGraphs [30]. It was collected from the project ITDK conducted by the Center for Applied Internet Data Analysis (CAIDA) [31]. This extraction employs the Filtered Recurrent Multi-level (FRM) algorithm, designed to capture the node agglutination patterns found on the Internet and ensure that the sizes of the subgraphs will be in a predefined range. IGraphs is especially useful for training DGGI. Furthermore, incorporating IGraphs allows the inclusion of realistic and comprehensive network scenarios in evaluating network protocols and mechanisms based on simulation and emulation.

The main contributions of this paper are:

- the introduction of a novel generator (DGGI) based on deep learning for the generation of intra-AS graphs that encodes not only the node degree distribution of training data but also their centrality, clustering, and assortativity;
- the presentation of a new dataset composed of real intra-AS graphs (IGraphs) extracted from the project ITDK.

Compared with existing generators for the Internet, DGGI improves the Maximum Mean Discrepancy (MMD) similarity index [32], [33], on average, by 84.4%, 95.1%, 97.9%, and 94.7% for assortativity, betweenness, clustering, and node degree, respectively. In the worst case, DGGI improves by 13.1% for assortativity, and in the best case, and 99.8% for clustering.

This paper is organized as follows. Section II presents related work focused on generators for Internet topology graphs; Section III shows the graph metrics used for the evaluation of the proposed solution; Section IV introduces the DGGI; Section V introduces the IGraphs dataset; Section VI describes the evaluation procedures adopted to validate the DGGI generator and discusses the results obtained. Section VII draws some conclusions and directions for future work.

II. Related Work

This section describes existing work related to graph generation to represent Internet-like topologies. Table I summarizes the characteristics of the reference work and shows how DGGI differs from other generators. The models are classified according to the applicability of their generated graphs to deep learning training. Generators classified as "Suitable for DL" (vide Table I) can synthesize realistic graphs for an arbitrary number of nodes, i.e., they are not restricted to generating large graphs with hundreds of thousands of nodes.

Most generators employ algorithms based on scale-free networks and power-law node degree distribution [7], [8], [9], [10], [13], [14], [15], [16], [17], [18]. The models in [8], [9], [10], [17] aim to generate graphs as a composition of subgraphs (structures), *e.g.*, AS nodes, core, and periphery.

The BA algorithm [12] is based on the preferential attachment property, *i.e.*, nodes with the highest node degree values tend to have new links attached. The Boston University Representative Internet Topology Generator (BRITE) [13] implements the BA algorithm to generate Internet graphs for inter-AS and intra-AS topologies.

A set of generators based on the BA algorithm has been proposed to introduce new strategies for the preferential attachment paradigm to enhance the ability to generate more realistic graphs. The Extended Barábasi-Abert (EBA) algorithm randomly modifies links beyond the preferential attachment [15]. The Bianco-Barábasi (BB) algorithm introduces a set of parameters (fitness weights) to specialize the preferential attachment mechanism [15]. In contrast, the Dual Barábasi-Abert (DBA) algorithm changes the number of links to be attached to new nodes in a random way [14]. The BB algorithm can reproduce the empirical power-law decays for Autonomous System (AS) graphs [6]. However, the generated graphs are general purpose, *i.e.*, they are not specific to Internet graphs.

Generators based on structure decomposition have been proposed to mitigate BA-based generators' limitations in reproducing the Internet graph's properties. The Simulates Internet graphs using the Core Periphery Structure (SICPS) [9] model partitions the Internet graphs into 16 structures that represent different statistical assumptions, including the power-law distribution. The SubNetwork Generator (SubNetG) [10] represents subnets and routers as a bipartite graph based on their power-law distribution. The Structure-Based Internet Topology gEnerator (S-BITE) [8] and Internet AS Graph (IAG) [18] generators use similar approaches to decompose the Internet into the core and periphery, each with a different distribution of power laws. The Jellyfish [17] generator captures the Internet's topology core (the ring) based on the assumption of a distribution by power law.

However, both structure-based and the BA-based generators rely on the assumption of power-law distribution, which restricts the generalization of node connectivity, since the decay parameter of a power-law distribution is insufficient to represent the topology diversity in the Internet [11].

An alternative proposal consists of employing different structuring procedures. The Orbis generator [7] creates Internet graphs by adopting dK-distributions as a criterion to maintain the correlation node degree of subgraphs of size d. Orbis uses the $1K$ and $2K$ distributions, which refer to the distributions of node degree and joint node degree, respectively. Although the dK-distributions attempt to unify a wide range of graph metrics [7], they focus only on the node degree, which can lead to a poor representation of other graph properties, such as clique formation and node centrality.

TABLE I
OVERVIEW OF RELATED WORK

Reference	Technique	Requirements	View	Suitable for DL	Validation
SICPS [9]	Multi-Structure Decomposition	Number of Nodes and Structure Statistical Properties	Inter-AS	No	First-Order Moments of Node Degree, Assortativity, and Clustering
SubNetG [10]	Scale-free and Hierarchical Decomposition	Number of Nodes and Component Power Law Coefficients	Intra-AS	No	First-Order Moments of Node Degree and Clique Sizes
S-BITE [8]	Scale-Free and Core-Periphery Decomposition	Number of Nodes and Core-Periphery Statistical Properties	Inter-AS	No	First-Order Moments of Node Degree, Clustering, Betweenness, Closeness, and Clique Size
Jellyfish [17]	Scale-Free and Ring Decomposition	Number of Nodes and Ring Power Law Coefficients	Inter-AS	No	First-Order Moments of Node Degree
IAG [18]	Scale-free and Hierarchical Decomposition	Number of Nodes	Inter-AS	No	First-Order Moments of Node Degree
Orbis [7]	dK-Series Preservation	Number of Nodes and dK-Series	Both	No	First-Order Moments of Node Degree and Betweenness
BRITE [13]	Barábasi-Abert	Number of Nodes and Preferential Attachment Coefficients	Both	Yes	First-Order Moments of Node Degree
DBA [14]	Dual Barábasi-Abert	Number of Nodes and Preferential Attachment Coefficients	–	Yes	–
BB [15]	Bianco-Barábasi	Number of Nodes and Preferential Attachment Coefficients	–	Yes	First-Order Moments of Node Degree
EBA [16]	Extended Barábasi-Abert	Number of Nodes and Preferential Attachment Coefficients	–	Yes	First-Order Moments of Node Degree
DGGI (our)	Deep Learning	Number of Nodes and DL Weights	–	Yes	Multi-Order Moments of Node Degree, Clustering, Betweenness, and Assortativity

– : No applicable

However, all of these generators mentioned above have been validated only by inspecting the first-order moments of the selected metrics (as indicated in Table I). Moreover, classical generators are often manually parameterized based on specific methodologies involving the inspection of Internet topologies. This process is time-consuming and requires a substantial manual overhaul of the model parameters. Orbis is the only exception since it implements an automatic procedure to adapt to the network scenario considered.

In contrast, DGGI generators do not depend on the power-law assumption, and the modeling is not focused only on the node degree. DGGI, which is based on deep learning (DL), can be trained for various network scenarios without substantial overhaul. Furthermore, our evaluation is not restricted to the first-order moments of graph properties; we explore higher-order moments using the MMD metric to quantify the similarity between graph distributions.

III. GRAPH METRICS

Four graph metrics are used to analyze the properties of the generated graphs: node degree, clustering coefficients, interdependence, and assortativity [34].

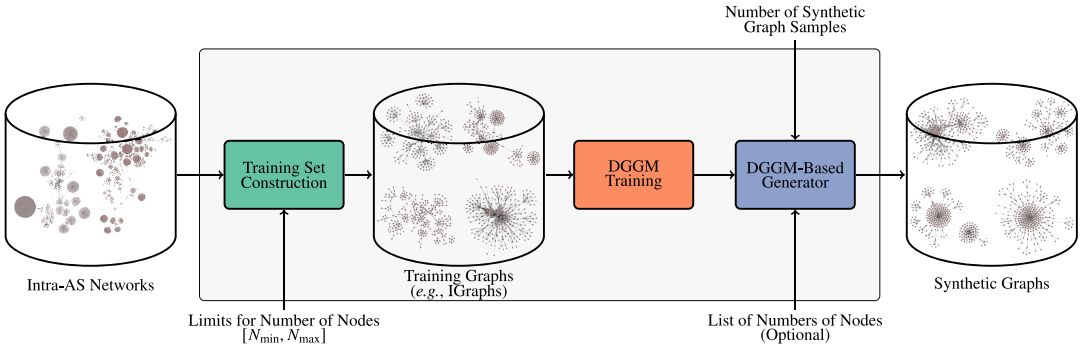


Fig. 1. The training pipeline for generative models tailored to the Deep Graph Generative Model (DGGM) is designed to synthesize Internet-like graph, which includes deriving a training dataset from limited samples of expansive networks and incorporating a layer for controlling both the quantity and number of nodes of the synthesized graphs. The instantiation of DGGI is based on three procedures: training set construction, DGGM training, and DGGM-based synthetic graph generation.

Let G be an undirected graph with N nodes and $A \in \{0,1\}^N$ be the adjacency matrix of the graph G. The node degree is the number of connections of a node, *i.e.*, the node degree of the i-th node is

$$\kappa_i = \sum_j^N A[i,j] \in \mathbb{N}, \tag{1}$$

in which $A[i, j]$ is the element on the i-th row and j-th column of the adjacent matrix.

The clustering coefficient indicates the tendency of a node to cluster with its neighbors, *i.e.*, the occurrence of density-connected regions in the graph. The clustering coefficient for the i-th node is defined as

$$C_i = 2\frac{l_i}{\kappa_i(\kappa_i - 1)} \in [0,1] \tag{2}$$

in which l_i is the number of edges between the neighbors of node i. Moreover, the global clustering coefficient can also be defined as

$$C = \frac{N_\Delta}{N_3} \in [0,1] \tag{3}$$

in which N_Δ is the number of triangles in the graph, and N_3 is the number of connected triads of nodes, *i.e.*, all sets of three nodes that are connected by either two or three undirected edges.

The betweenness coefficient measures the importance of a node in a graph. It quantifies the number of times a node acts as a bridge along the shortest path between two other nodes. This measure is essential in understanding the role of a node in communication networks. Nodes with high betweenness centrality are critical for the transfer of information or resources within the network. They often serve as bottlenecks or hubs that facilitate communication. Formally, the betweenness for the i-th node can be defined as

$$\mathcal{B}_i = \sum_{\substack{p \neq i \neq q \in \{1,\ldots,N\} \\ p \neq q}} \frac{\sigma_i(p,q)}{\sigma(p,q)} \in [0,1] \tag{4}$$

in which $\sigma_i(p, q)$ is the number of shortest paths between the p-th and q-th nodes that pass through the i-th node, and $\sigma(p, q)$ is the total number of shortest paths between p and q.

The assortativity coefficient measures the correlation between the degrees of connected nodes. A positive degree assortativity coefficient indicates that nodes tend to connect with other nodes that have a similar degree. In contrast, a negative coefficient suggests that the high-degree nodes are more likely to connect with the low-degree nodes. So, it indicates whether or not the growth of the network follows the preferential attachment pattern. Assortativity is a scalar measure defined as

$$r = \frac{\sum_k e_{kk}}{\sum_k \alpha_k \beta_k} \in [-1,1], \tag{5}$$

where k is the node degree of graph G, $\alpha_k = \sum_{k'} e_{kk'}$, $\beta_k = \sum_{k'} e_{k'k}$, and $e_{kk'}$ is the number of edges of nodes with degree k and nodes with degree k'.

IV. Deep-Generative Graphs for the Internet (DGGI) Model

Figure 1 illustrates the three procedures used to instantiate the DGGI generator: the construction of the training set based on applying the FRM algorithm, the training of the deep-generative graph model, and the generation of synthetic graphs.

For constructing the training set, the input is a set of samples from large intra-AS networks, with lower and upper bounds reflecting the number of nodes. The final training dataset contains only graphs with a given number of nodes in the defined range.

The Deep Graph Generative Model (DGGM) is then trained using the training graphs created during this first procedure. The final procedure then involves using the trained model to synthesize intra-AS graphs based on two parameters: an optional list that defines the number of nodes and the number of graph samples. When not specified, the defined range is that of the construction of the training set.

The implementation of each procedure is described below.

A. Training Set Construction

To extract subgraphs with a size bounded by a predefined limit, we introduce the Filtered Recurrent Multi-level (FRM)

algorithm, which employs the multilevel algorithm [35] recursively, followed by using a filter based on betweenness centrality to avoid a single-star topology. The multilevel algorithm aims to define the communities that maximize the node's local contribution to the overall modularity score. This score measures the ratio between the density of intracommunity and intercommunity links. For a graph with $|E|$ edges, the modularity can be defined as

$$Q = \sum_{p_i \in \mathcal{P}} \left[\frac{S_{\text{in}}^{p_i}}{2|E|} - \left(\frac{S_{\text{tot}}^{p_i}}{2|E|} \right)^2 \right] \tag{6}$$

where $\mathcal{P}$ is the set of graph communities, p_i is the i-th community, $S_{\text{in}}^{p_i}$ is the number of edges into community p_i, and $S_{\text{tot}}^{p_i}$ is the number of edges incident to the nodes in the community p_i.

The multilevel algorithm operates in two steps. In Step 1, each node in a graph is assigned a distinct community. For each node, it assesses the gain in modularity by relocating the node to the community of another node. The node is placed in the community that yields the maximum positive modularity gain; if the gain is negative, the node remains in its current community.

Step 2 involves constructing a new graph using the communities identified in Step 1 as nodes. Nodes are formed by merging all nodes within a community into a single node. Links between nodes of the same community result in self-loops for the community in the new graph. Then, using this new graph, the algorithm iterates through Step 1 until no further improvement in modularity can be achieved.

The recursive application of the multilevel algorithm may lead to graphs with a single hub (graph with a star topology) due to the high disparity between intra-hub and inter-hub links observed in subgraphs with many hubs and low clustering coefficients. All subgraphs defined by a single hub are discarded using the aforementioned filter to avoid redundant occurrences of hubs in the training dataset.

Algorithm 1 presents the FRM algorithm used to construct the training set. The lists defined in the first two lines control the subgraph extraction process. The list *to_process* contains the graphs that must be split into smaller subgraphs, while the list *training_set* contains the subgraphs that define the training set.

The first condition on Line 6 checks whether the graph in the loop has a node count greater than the upper limit (N_{max}). If this condition holds, clusters are attempted to be extracted from the graph under consideration using the multilevel algorithm, and the loop continues. The condition in Line 8 ensures that the subgraphs are pushed to *to_process* only if the multilevel algorithm can split the graph. Otherwise, they are ignored.

In contrast, if the node count does not exceed N_{max}, the second condition in Line 12 comes into play. This condition ensures that the graph under consideration meets the criteria of having a node count greater than the lower limit (N_{min}) and is not structured as a single star topology, *i.e.*, a topology that has more than one node with a betweenness coefficient greater than zero.

Algorithm 1 Filtered Recurrent Multi-level (FRM)

Require: A graph G, the minimum $(N_{\min})$ and the maximum $(N_{\max})$ of the number of nodes.

```
1:  to_process = [G]
2:  training_set = []
3:  while |to_process| > 0 do
4:      G = to_process.pop()
5:      N = number_of_nodes(G)
6:      if N > N_max then
7:          clusters = multilevel(G)
8:          if |clusters| > 1 then
9:              to_process = concat(to_process, clusters)
10:         end if
11:     else
12:         if N > N_min and |[B_1, ..., B_N] > 0| > 1 then
13:             training_set = concat (training_set, clusters)
14:         end if
15:     end if
16: end while
17: return training_set
```

The FRM algorithm is a recursive algorithm designed to stop when all graph components fall within the range of $[N_{\min}, N_{\max}]$. The best-case scenario occurs when a single call of the multilevel algorithm successfully places all components within this range. In this optimal case, the algorithm complexity is $O(N)$ since the multilevel algorithm has a linear complexity as a function of the number of nodes [35].

In contrast, the worst-case scenario arises when each call to the multilevel algorithm divides the original graph into two components, each having half the number of nodes of the original graph. The termination condition for the algorithm FRM is met when all graph components satisfy the node number constraints, *i.e.*, are in the range of $[N_{\min}, N_{\max}]$. In this worst-case scenario, the algorithm's complexity is determined by the structure of a binary tree, where each leaf represents a graph component with nodes in the range $[N_{\min}, N_{\max}]$. The complexity in the worst case is $O(2(1 - 2^h)N)$, since the number of nodes in a binary tree of height h is the sum of a finite geometric series whose elements are the sum of nodes in each level of the tree. The tree height can be estimated as

$$N_{\min} \leq \frac{N}{2^h} \geq N_{\max} \tag{7}$$

$$\frac{N}{N_{\min}} \geq 2^h \leq \frac{N}{N_{\max}} \tag{8}$$

$$\log_2\left(\frac{N}{N_{\min}}\right) \geq h \leq \log_2\left(\frac{N}{N_{\max}}\right), \tag{9}$$

since the total number of leaves in a binary tree is 2^h.

If the multilevel algorithm fails to generate clusters, FRM will stop the loop in Line 3 and return the training set, regardless of its content, due to the condition in Line 8. In contrast, convergence is guaranteed by the depletion of the stack of graphs awaiting processing, a state achieved when

the multilevel algorithm ceases to identify new clusters or all identified clusters contain fewer nodes than $N_{\max}$.

B. Deep Graph Generative Model Training

DGGI uses the recently proposed GraphRNN [33] as the deep-generative graph model. This model is a general-purpose data-driven generator of synthetic graphs based on deep learning. GraphRNN is acknowledged as the current state-of-the-art method for graph generation, demonstrating the generation of synthetic graphs that exhibit greater realism compared to other DGGMs, such as GraphVAE [36] and Deep Generative Model of Graphs (DeepGMG) [37].

The architecture of GraphRNN comprises two different hierarchical Gate Recurrent Units (GRUs) [38], one to embed the graph representation and the other to predict the new connections for each node. A Gate Recurrent Unit (GRU) is a variation of recurrent neural networks specialized in embedding relations established by long-ordered sequences of tensors into a state, a vector with a pre-defined dimension (so-called latent dimension). GraphRNN maps the graph generation into a sequential procedure based on the edges added to each node. In order to make this feasible, GraphRNN establishes a node order to process all graphs during training.

There is no unique node order to represent a graph using a sequence of nodes. The number of possible representations of this sequence-based graph is $N!$ with N being the number of nodes [33]. GraphRNN uses the pre-defined node order to reduce the number of possible node permutations, improving the learning efficiency [33]. GraphRNN uses the node order established by the Breadth First Search (BFS) algorithm since different node permutations can be mapped onto a unique node order [33].

Formally, let G be a graph with N nodes defined by a given order. A sequence to represent the graph G is described as $s = (s_1, \ldots, s_N)$, with $s_i \in \{0,1\}^{i-1}$ being the vector representing all connection between the node i and the remaining $i-1$ nodes. Assuming the BFS order for the nodes, the dimension of vector $s_i\ \forall i$ can be bounded by a fixed number M [33], the transient dimension. Therefore, a vector $s_i\ \forall i \in \{2, \ldots, N\}$ can be redefined as

$$s_i = [A[\max(i, i-M), i], \ldots, A[i-1, i]], \tag{10}$$

where $A[i, j]$ is the element of the i-th row and j-th column of the adjacency matrix of the graph G.

Algorithm 2 outlines how GraphRNN synthesizes a graph. The loop in Line 2 determines all the connections for each new node. These connections are defined for all $N_{\max}$ nodes (as defined in Section IV-A). In Line 3, the state h_g is determined by the GRU g using the connections of i-th node and the previous state h_g. The loop in Line 5 determines the $\min(i-1, M)$ connections of the $(i+1)$-th node, in which i nodes have already been added to the graph. Finally, each j connection is determined by the second GRU f using the current h_g state and the j-th connection of the i-th node. The complexity of Algorithm 2 is $O(\min(i-1, M)N_{\max}C_g + N_{\max}C_f)$, where C_g and C_f are the respective complexities for GRU layers f and g.

Algorithm 2 GraphRNN

Require: The maximum number of nodes $N_{\max}$, the latent dimension L, the transient dimension M, two GRU layers, $g : (\mathbb{R}^M, \mathbb{R}^L) \mapsto \mathbb{R}^L$ and $f : (\mathbb{R}, \mathbb{R}^L) \mapsto \mathbb{R}$, the initial state h_g, and the start and end tokens, $SOS \in \mathbb{R}^M$ and $EOS \in \mathbb{R}^M$.

1: $s_1 = SOS$ and $i = 1$
2: **for each** $i \in \{1, \ldots, N_{\max}\}$ **do**
3: $\quad h_g = g(s_i, h_g)$
4: $\quad s_{i+1} = s_i$
5: $\quad$ **for each** $j \in \{1, \ldots, \min(i-1, M)\}$ **do**
6: $\quad\quad h_f = f(s_{i+1}[j], h_g)$
7: $\quad\quad s_{i+1}[j] = h_f$
8: $\quad$ **end for**
9: **end for**
10: **return** $\{s_1, \cdots, s_{N_{\max}}\}$

C. Synthetic Graph Generation

The final procedure synthesizes intra-AS graphs using the trained GraphRNN model. Two parameters are required to generate graphs: the number of synthetic graphs and, optionally, a list of the number of nodes.

Synthetic graphs are created using the output of Algorithm 2 given a trained GraphRNN. In order to define a graph, the vectors $\{s_1, \ldots, s_{N_{\max}}\}$ provided by the trained GraphRNN are transformed into an adjacency matrix A. This transformation requires mapping the vector values to be 0 or 1 since $s_i \in [0,1]^M\ \forall i$. A threshold τ is used to implement the mapping. The algorithm assigns 1 if the value is higher than τ and 0 otherwise.

Using a fixed value of τ will produce the same set of graphs for all runs of Algorithm 2 due to the fixed weights of GraphRNN. To prevent such reproduction, the threshold τ is sampled from a uniform distribution $\mathcal{U}(]0,1[)$ each time the M edges of a node are defined, resulting in N random parameters for each synthetic graph, with N representing the number of nodes. This sampling ensures that a unique set of graphs is generated for each run of the algorithm, highlighting the crucial role of the uniform distribution in the graph generation process.

Given the statistical nature of both deep learning models and the threshold τ, the adjacency matrix resulting from the former process does not necessarily represent a connected graph. The connected subgraphs provided by the adjacency matrix provide the synthetic graph.

Algorithm 3 outlines how the generator is defined using a pre-trained GraphRNN. In Lines 6 to 8, the threshold, an empty adjacent matrix, and the node states are defined, respectively. The loop in Line 10 maps each vector state s_j to the proper column of the adjacency matrix using the threshold τ and the transient dimension M (defined in Section IV-B). The loop in Line 16 extracts all connected subgraphs from the adjacency matrix. Only the subgraphs with a certain number of nodes included in the list L are considered. The algorithm stops when the number of synthetic graphs is greater than or equal to T. The generation of each graph has complexity bounded

Algorithm 3 Graph Generator

Require: A trained GraphRNN $\mathcal{F}$, the number of synthetic graphs T, and, optionally, the list of numbers of nodes L, the minimum $(N_{\min})$ and the maximum $(N_{\max})$ of the number of nodes, and the transient dimension M.

```
graphs = []
if L = ∅ then
    L = {N_min, ..., N_max}
end if
while |graphs| < T do
    τ ~ U(]0, 1[)
    A_k = {0}^(N_max × N_max)
    {s_1, ..., s_{N_max}} = F()
    for each s_j ∈ {s_1, ..., s_{N_max}} do
        for each i ∈ {max{1, j − M}, j − 1} do
            if s_j[i] ≥ τ then
                A_k[i, j] = 1
            else
                A_k[i, j] = 0
            end if
        end for
    end for
    for each G = connected_subgraphs(A_k) do
        if number_of_nodes(G) ∈ L then
            graphs.push(G)
        end if
    end for
end while
return graphs
```

by $O(C_{\mathcal{F}}+N_{\max}^2)$ and $O(C_{\mathcal{F}}+N_{\max}^2+M^2-N_{\max}M)$, where $C_{\mathcal{F}}$ is the complexity of GraphRNN.

If the list of node numbers L includes values beyond the predetermined range $[N_{\min}, N_{\max}]$ established by the training dataset, the convergence of Algorithm 3 cannot be assured. This limitation arises due to the specialization of the GraphRNN model, which was trained specifically to produce graphs featuring a node count confined within the specified bounds of $[N_{\min}, N_{\max}]$.

V. The Internet Graphs (IGraphs) Dataset

This section describes the construction and properties of the proposed intra-AS graph dataset, IGraphs. IGraphs is used as the training graphs in Figure 1. Specifically, IGraphs is used to train GraphRNN defining the intra-AS generator DGGI.

A. The IGraphs Construction

The construction of IGraphs is based on applying FRM (Section IV-A) over a cross-section of the Internet with more than 90 million nodes provided by CAIDA through the ITDK repository. This repository stores the historical router-level topologies, providing the IPv4/v6 traces, the router-to-AS assignments, the router geographic location, and the DNS for all observed IP addresses.

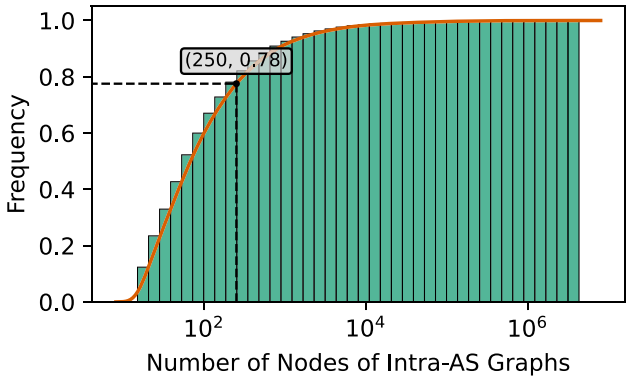


Fig. 2. Cumulative distribution of the numbers of nodes of intra-AS graphs.

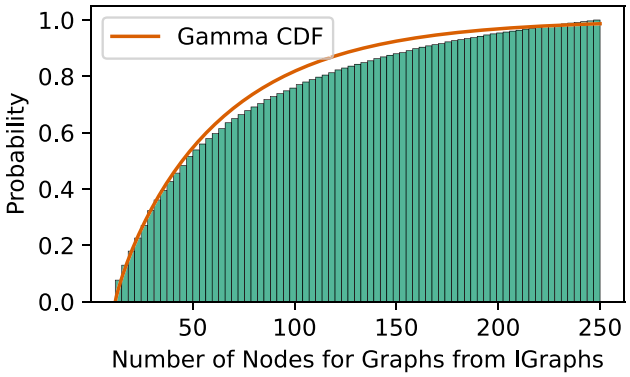


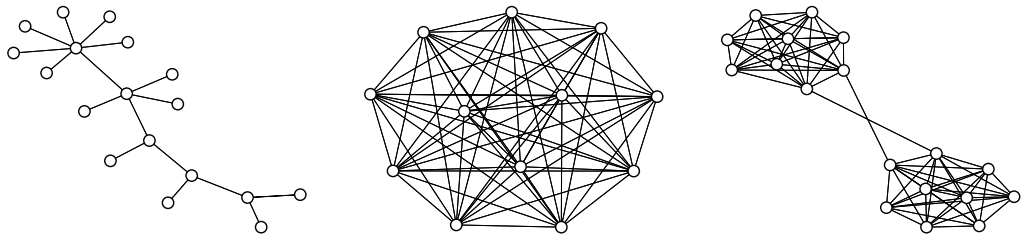
Fig. 3. The cumulative distribution of the numbers of nodes of IGraphs. The orange curve represents the CDF of gamma with parameters fitted to the presented cumulative distribution.

This Internet cross section is partitioned into AS subgraphs using router-to-AS tables and IPv4 links. The links are pre-processed to extract edges, as multiple nodes can share a single link, resulting in multiple edges per link. Subsequently, all edges where the predecessor and successor are within the same AS are classified as intra-AS edges, thus delineating the AS topologies.

The AS topologies are simplified to emphasize the router relations. Multiedges (edges with the same pair of predecessor and successor) and self-edges (edges with the predecessor equal to the successor) are discarded. Then, the IGraphs graphs are created after defining the intra-AS edges.

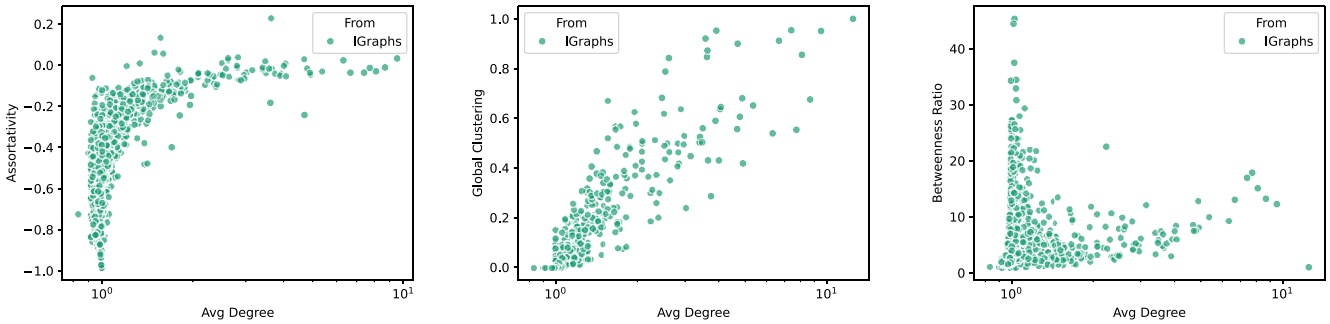
Each AS topology is used to extract the graphs that will compose the IGraphs dataset. The number of nodes in the IGraphs graphs ranges from 12 to 250. These limits have been defined based on the number of nodes of graphs often used to evaluate recently proposed graph-based data-driven models in communication networks [3], [5], [10]. However, FRM can be used for other ranges.

Figure 2 illustrates the distribution of the number of nodes in AS graphs, indicating that 22% of the ASes contain more than 250 nodes. FRM was employed to obtain intra-AS graphs with a node count within the desired range. The resultant dataset (IGraphs) comprises 90,326 graphs, each containing between 12 and 250 nodes. The node count distribution is depicted in Figure 3.



Assortativity	-0.2260	-0.0693	-0.0278
Global Clustering	0.0000	0.9918	0.9403
Betweenness Ratio	2.6990	1.0455	10.6655

Fig. 4. Examples of graph samples from IGraphs presenting different graph metrics, evaluated in terms of their betweenness ratio, assortativity, and global clustering. The graph on the left does not contain triangles. This leads to a clustering coefficient equal to zero and a low assortativity score. The graph in the middle is a densely connected graph. In this case, the clustering coefficient is high, while the assortativity score is low. The graph on the right contains two hubs. In this case, the betweenness ratio score is high.



(a) Assortativity (b) Global Clustering (c) Betweenness Ratio

Fig. 5. The scattering of graph properties against the average node degree for IGraphs datasets. Figures (a) and (b) present the scattering of global assortativity and clustering, respectively; (c), shows the scattering of the ratio between the maximum and the average of the betweenness coefficients.

B. IGraphs Properties

The analyses presented in this section are based on three graph metrics: the coefficients for assortativity, clustering, and betweenness. For the latter, the ratio between the maximum and average betweenness coefficients. This ratio encodes the number of hop occurrences in each graph [39]. Figure 4 illustrates the values calculated associated with the three metrics for three graph samples collected from IGraphs.

Figure 5 shows the scattering of the graph metrics used in relation to the average node degree calculated using the approach adopted in [39]. The global clustering and assortativity scores were calculated as defined in Equations (3) and (5). The maximum and average values were calculated for the betweenness ratio according to Equation (4).

Fig. 5(a) shows that graphs from IGraphs with a higher average node degree tend to present null assortativity, and graphs with nodes with a high node degree do not follow the precepts of preferential attachment, unlike graphs with small node degrees. In this case, the negative assortativity indicates an inverse preferential attachment prone.

Figure 5(b) shows the occurrences of density subgraph networks in IGraphs. For graphs with a low average node degree, there is a trend to a monotonic increase in global clustering, which is not verified for graphs with a high average node degree. These graphs present a greater dispersion, which indicates that nodes are not necessarily prone to triangle formation when the node degree increases.

Fig. 5(c) suggests that a low average node degree leads to a greater probability of hub occurrences. At the same time, the increase in average node degree does not imply a large occurrence of hubs, *i.e.*, the centrality tends to be homogeneous among all nodes.

IGraphs comprises intra-AS graphs that take advantage of novel possibilities to analyze solutions in communication networks. The 90,326 graph samples from IGraphs allow training data-driven models based on real Internet topologies instead of augmentation procedures over a few samples of real networks, *e.g.*, from the Topology Zoo [39] dataset.

Training using IGraphs can also improve the generalization capability, preventing overfitting since IGraphs presents large topological variability.

Typically, evaluations of network mechanisms are based on simulations or emulations running over simple topologies (*e.g.*, grids, stars) or on a few samples from real topologies (*e.g.*, NSF, Geant2, and Germany50) [3], [5]. On the other hand, introducing IGraphs opens opportunities to perform more robust evaluations based on the diversity of real-world topologies.

VI. Evaluation of DGGI

This section describes the procedure followed to assess the effectiveness of DGGI, instantiated as described in Section IV. The assessment consists of quantifying to the extent that the

generated synthetic topologies differ from real-world intra-AS graphs.

Section VI-A introduces the Maximum Mean Discrepancy (MMD) metric, which estimates whether two samples are modeled from the same distribution [32]. Section VI-B describes the training procedures, while Section VI-C describes the baseline generators considered in our study. Finally, the results are presented and discussed in Section VI-D.

A. Maximum Mean Discrepancy (MMD)

Let P and Q be two distributions defined in a metric space X; $\mathcal{H}$ be a reproducing kernel Hilbert space (RKHS) with a kernel κ; and ϕ be a function that maps X to $\mathcal{H}$. MMD is defined as

$$\begin{aligned}\mathrm{MMD}(P,Q) &:= \left\| \mathop{E}_{x\sim P}\phi(x) - \mathop{E}_{y\sim Q}\phi(y) \right\|_{\mathcal{H}} \\ &= \mathop{E}_{\substack{x\sim P\\ x'\sim P}}\kappa\left(x,x'\right) - 2\mathop{E}_{\substack{x\sim P\\ y\sim Q}}\kappa(x,y) + \mathop{E}_{\substack{y\sim Q\\ y'\sim Q}}\kappa\left(y,y'\right), \end{aligned} \tag{11}$$

which satisfies the metric properties, such as $\mathrm{MMD}(P,Q) = 0$ iff $P = Q$ [32].

The MMD kernel used is defined as follows:

$$\kappa_{\mathcal{W}}(P,Q) \ = \exp\left(\frac{\mathcal{W}(P,Q)}{\sigma^2}\right) \tag{12}$$

in which $\mathcal{W}$ is the Wasserstein distance, and σ is a free parameter similar to a Radial Basis Function (RBF) kernel. This function maintains the MMD assumptions since it induces a unique RKHS [33], and all statistical moments, which can be verified by the Taylor expansion of $\kappa_{\mathcal{W}}$ [33].

The MMD for all graph metrics (node degree, clustering, betweenness, and assortativity) were estimated by bootstrap sampling, given the computational cost of MMD evaluation. This procedure consists of sampling 500 graphs from the IGraphs dataset with replacement and assessing the MMD (for all graph metrics) using these graphs with other 500 synthetic graphs created by either baselines or our DGGI generator. This bootstrap evaluation was repeated 100 times, allowing the establishment of a confidence interval associated with each MMD value computed for each graph metric.

B. Training the DGGI Generator

IGraphs was divided into three distinct parts: training, validation, and testing. All sample graphs from IGraphs were shuffled, and 70% were reserved for the training set, while the remaining graphs were divided equally for the validation and test sets. This partition led to 63.229, 13.547, and 13.547 graphs for training, validation, and testing, respectively.

The generator DGGI learns the conditional distribution that models the generation of links of the graphs in the training data, *i.e.*, given a previous state for graph representation, the generator predicts a new link. Binary cross entropy [40] is used as the loss function since the prediction of links is mapped to a binary classification problem to determine if a link exists.

The machine used for training had an i7-9700 CPU and a Quadro RTX 6000 GPU. The training procedure consisted of 500 epochs using the Adaptive Moment Estimation (ADAM) optimizer [41] with an initial learning rate of 0.003, decaying by a factor of 0.3 when the training reaches 300 epochs. Backpropagation for each mini-batch composed of 40 graphs sampled from the training set was evaluated using uniform bootstrap sampling. The best model was determined based on the best MMD value obtained for the validation set, considering the node degree distributions.

Fig. 6 shows the distributions of all the graph metrics mentioned. Based on these distributions, it can be inferred that the generator DGGI reproduces the distribution of the test set accurately for all the metrics assessed. Figures 6(a), 6(c), 6(e), and 6(g) indicate that those based on the BA method (*i.e.*, BRITE, BB, EBA, DBA) do not reproduce the test set distribution for all metrics.

C. Baselines

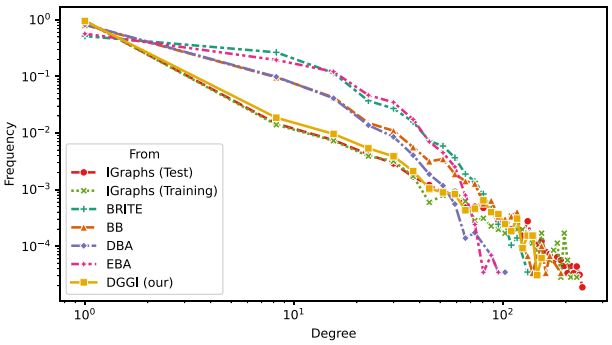
The baseline generators adopted were divided into two groups: generators based on the BA algorithm, BRITE, BB, EBA, and DBA, and those based on the structure-based models of SubNetG, S-BITE, IAG, and Orbis.

The BA-based models were configured using commonly used parameter values [6], [11], [13]. Each new node establishes two new links after preferential attachment. In BRITE, these links connect the new nodes with existing ones. EBA was configured to behave as BRITE 50% of the time, while 25% of the links were added to the existing nodes, and for the other 25%, two known links were rewired. DBA uses two BA models simultaneously; 35% of the time, one link was added to any new node instead of two links.

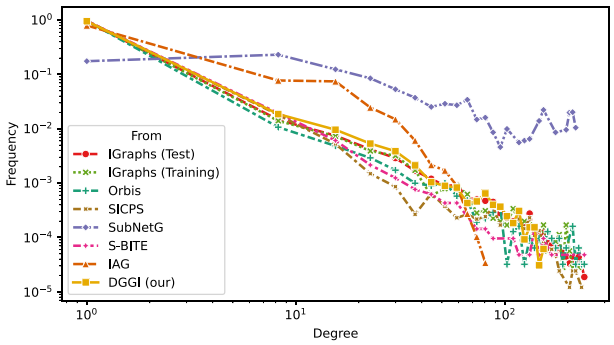
The number of nodes of the synthetic graphs generated by BA-based models was randomly determined. In order to improve the similarity between these graphs and IGraphs graphs, those generated by BA-based models were adjusted to have several nodes drawn from a customized gamma distribution. This distribution was tailored by fitting it to the distribution of the number of nodes of IGraphs graphs, as illustrated in Fig. 3. The fitting process consisted in the employment of Maximum Likelihood Estimation (MLE), resulting in a local optimal gamma distribution expressed concerning x as $(y^{a-1}\exp(-y)/(s\Gamma(a))$, with $a = 0.852$, $s = 59.64$, $y = (x - m) / s$, and $m = 11.99$.

On the other hand, for structure-based models, the configurations used followed the parameters suggested in [7], [8], [9], [10]. For S-BITE and SubNetG, the parameters were determined using topologies provided by the Internet Research Lab (IRL)-based dataset [8], [10]. The joint distribution of node degrees required by Orbis was extracted from the CAIDA AS topology (Section V). Moreover, IAG did not require any further configuration.

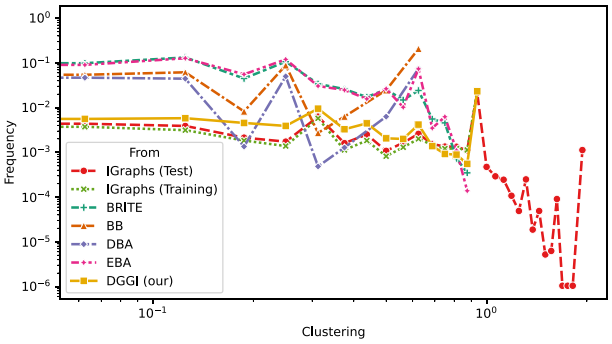
Unlike BA-based models, S-BITE, SubNetG, and Orbis models were designed to generate graphs in the range of hundreds of thousands of nodes. The FRM algorithm was used to extract small subgraphs from the large synthetic graphs provided by those baselines (Section V) since graphs with hundreds of nodes are expected.



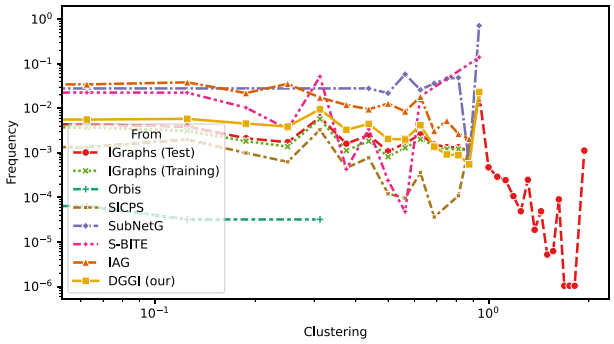
(a) BA-based baselines.



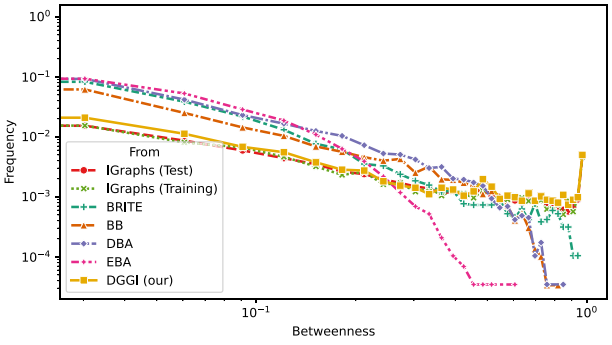
(b) Structure-based baselines.



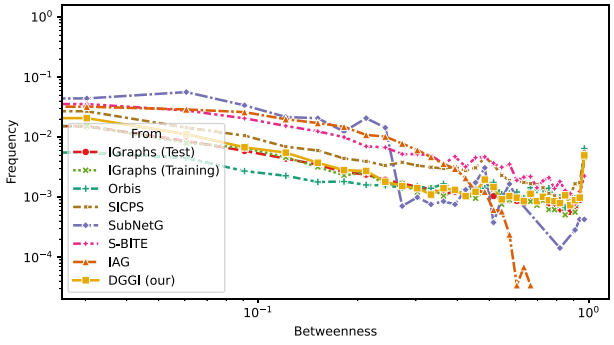
(c) BA-based baselines.



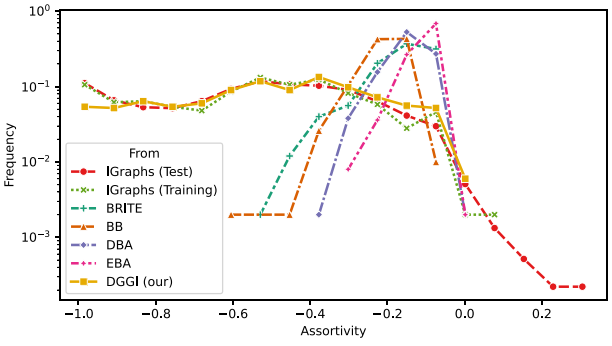
(d) Structure-based baselines.



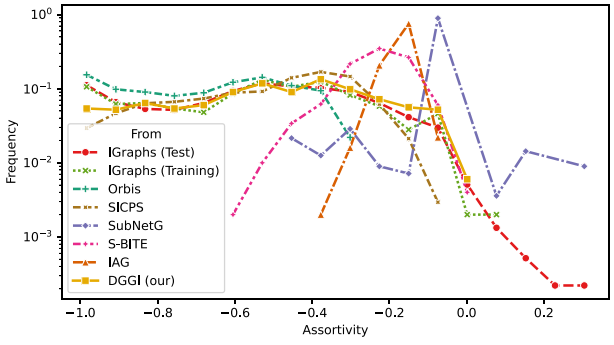
(e) BA-based baselines.



(f) Structure-based baselines.



(g) BA-based baselines.



(h) Structure-based baselines.

Fig. 6. The figure shows the frequency distribution for the occurrence of each assessed graph metric. For comparison, all graphs present the distribution for CAIDA and for our model, DGGI, and the baselines are organized in two groups, BA and structure-based.

D. Results and Discussion

We aim to assess the similarity of the graphs synthesized by the DGGI generator considering the test set extracted from IGraphs (Section VI-B). The same comparison was performed for the baseline generators. The similarity is quantified using the first and higher moments of the following

TABLE II
ACHIEVED AVERAGE MMD FOR DIFFERENT GRAPH METRICS

IGraphs	MMD Assortativity	MMD Betweenness	MMD Clustering	MMD Node Degree
Training Sampling	7.09e−03 ± 6.01e−03	4.23e−04 ± 3.96e−04	7.15e−04 ± 6.20e−04	2.25e−03 ± 1.59e−03
Generator	**MMD Assortativity**	**MMD Betweenness**	**MMD Clustering**	**MMD Node Degree**
BB	1.58e+00 ± 4.03e−02	4.67e−02 ± 2.25e−03	9.62e−01 ± 2.64e−02	8.95e−01 ± 1.26e−02
BRITE	1.70e+00 ± 3.22e−02	5.68e−02 ± 2.57e−03	4.54e−01 ± 1.44e−02	5.62e−01 ± 1.26e−02
DBA	1.75e+00 ± 3.28e−02	7.99e−02 ± 2.77e−03	4.16e−01 ± 2.32e−02	1.06e+00 ± 1.33e−02
EBA	1.85e+00 ± 1.95e−02	1.04e−01 ± 4.16e−03	8.54e−01 ± 2.26e−02	6.86e−01 ± 1.26e−02
IAG	1.69e+00 ± 3.36e−02	7.87e−02 ± 3.39e−03	3.45e−01 ± 1.75e−02	8.51e−01 ± 1.33e−02
Orbis	3.49e−01 ± 5.63e−02	4.59e−03 ± 7.33e−04	5.02e−02 ± 7.29e−03	2.35e−02 ± 3.04e−03
S-BITE	1.45e+00 ± 5.17e−02	1.19e−01 ± 8.64e−03	6.24e−01 ± 2.49e−02	3.23e−01 ± 1.69e−02
SICPS	8.70e−02 ± 2.18e−02	4.17e−02 ± 5.50e−03	2.76e−02 ± 5.37e−03	6.27e−02 ± 9.18e−03
SubNetG	1.68e+00 ± 3.66e−02	1.03e−01 ± 6.49e−03	1.54e+00 ± 1.76e−02	5.95e−01 ± 1.23e−02
DGGI (our)	**7.56e−02 ± 3.26e−02**	**1.31e−03 ± 8.77e−04**	**2.80e−03 ± 1.54e−03**	**6.82e−03 ± 2.17e−03**

four graph metrics: node degree, clustering, betweenness, and assortativity.

Figure 6 illustrates the distributions for the considered graph metrics. It displays results for two sets of baselines, BA-based and structure-based generators. The distributions for the real graphs (sampling from training and test datasets) and the graph synthesized by DGGI are also provided to simplify the comparison. However, the distributions in Fig. 6 allow only a visual comparison and give a limited notion of mean, variance, and other higher moments. The MMD was used to concisely represent the differences between the test and training sets of these statistical moments to provide a more significant comparison.

For structure-based baselines, Figures 6(b), 6(d), 6(f), and 6(h) show that neither IAG nor SubNetG succeeded in reproducing the distributions of the test and training sets. Orbis and SICPS can visually decrease the overall distance concerning the distributions of the test and training sets for the node degree, betweenness, and assortativity. However, Orbis does not reproduce the clustering distribution accurately. accurately. S-BITE does not visually reproduce the right tail of distributions of the test and training sets for the clustering, betweenness, and assortativity metrics.

In contrast, Figures 6(a), 6(c), 6(e), and 6(g) depict the distributions for BA-based baselines. All BA-based generators exhibit similar behavior regarding node degree, with baseline distributions shifted relative to real ones (training and test sampling from IGraphs), except for the distribution tails. We aim to assess the similarity of the graphs synthesized which BA-baselines BA-baselines reasonably reproduce. For betweenness, the behavior of the baselines mirrors that observed for node degree; however, only BRITE accurately reproduces the right tail of the distribution. Lastly, the BA baselines prove inadequate in reproducing the observed distributions for clustering and assortativity metrics.

Table II shows the MMD values assessed for the four graph metrics to quantify the similarity between the baseline distributions and the real distribution defined by the sample from the test set. Orbis and SICPS outperform other baselines regarding MMD values across all graph metrics, suggesting their ability to replicate higher-order statistical properties of real intra-AS topologies. Generally, baselines utilizing BA model exhibit inferior performance in reproducing intra-AS topology, as evidenced by their lower MMD values than their structure-based counterparts. However, an exception is noted with BRITE, despite its status as the earliest generator, as it surpasses all baselines except Orbis and SICPS in replicating the four metrics of high-order statistical properties. Furthermore, Table II presents the MMD values for the training sampling, which can serve as a reference due to the sampling of both the training and the test sets from the same population, corroborated by the low MMD values observed in the training sampling.

The node degree, indicative of the number of adjacent nodes, exhibits a notable correlation with betweenness centrality owing to considering neighboring nodes in determining shortest paths. This correlation is depicted in Figure 6, wherein the generators consistently manifest analogous errors in estimating node degree and betweenness centrality. Notably, baseline methods, particularly Orbis, SICPS, and BRITE, tend to synthesize network topologies exhibiting similar sets of shortest paths. The distribution of betweenness coefficients offers insight into the prevalence of hub nodes within a topology. As illustrated in Figure 6(e), BRITE-synthesized graphs exhibit lower frequencies of large betweenness coefficients compared to real-world counterparts from training and testing datasets, indicating a higher incidence of hubs in actual intra-autonomous System (AS) topologies. Conversely, Orbis and SICPS demonstrate comparable hub frequencies to real-world datasets, attributed to

TABLE III
ACHIEVED AVERAGE IMPROVEMENTS USING DGGI

IGraphs	Assortativity	Betweenness	Clustering	Node Degree
Training Sampling	-967.05%	-210.81%	-291.14%	-202.99%
Baseline	**Assortativity**	**Betweenness**	**Clustering**	**Node Degree**
BB	95.22%	97.19%	99.71%	99.24%
BRITE	95.56%	97.68%	99.38%	98.79%
DBA	95.68%	98.35%	99.33%	99.35%
EBA	95.91%	98.74%	99.67%	99.01%
IAG	95.54%	98.33%	99.19%	99.2%
Orbis	78.31%	71.35%	94.43%	70.93%
S-BITE	94.78%	98.9%	99.55%	97.89%
SICPS	13.08%	96.85%	89.86%	89.11%
SubNetG	95.49%	98.72%	99.82%	98.85%

their ability to reproduce the right tail of the betweenness distribution.

Clustering, which denotes the prevalence of triadic relationships among nodes (Section III) manifests itself in densely interconnected regions within a network. However, none of the baseline models successfully reproduce the clustering coefficients observed in real intra-AS topologies, as evident from Figures 6(c) and 6(d). Furthermore, Table II reports clustering MMD values significantly deviating from zero for baseline methods, indicating a substantial dissimilarity with real-world clustering patterns, up to 20 to 30 times worse concerning clustering MMD values for training sampling. Networks generated by baseline methods tend to exhibit dense node regions inconsistent with real intra-AS topologies.

Assortativity, which indicates the correlation between connections of nodes that share similar neighborhood sizes, still needs to be attainable for most baseline models, with Orbis and SICPS being exceptions. As illustrated in Figure 6 and Table II, baseline models fail to replicate the assortativity coefficients observed in real intra-AS graphs, implying a divergence from the attachment tendencies characteristic of actual network formations. In contrast, the Orbis and SICPS models demonstrate a reasonable approximation of this attachment tendency, aligning more closely with the observed network assortativity patterns.

DGGI outperforms all baselines in terms of the realism of its synthetic graphs. For all metrics, node degree, betweenness, clustering, and assortativity, DGGI substantially reduces overall dissimilarity in terms of the distributions observed in both training and test sets, as illustrated in Figure 6. However, none of the generators, including DGGI, accurately replicate the right tails of the distributions for clustering and assortativity (Figures 6(c), 6(d), 6(g), and 6(h)). This inability of DGGI to reproduce these right tails could be attributed to potential overfitting, given that these right tails manifest exclusively in the test set. Furthermore, Table II shows that DGGI surpasses all baselines, even considering the high-order statistical properties assessed through MMD.

Table III summarizes the significant improvements achieved by DGGI, presenting the average improvements of DGGI relative to the baselines. On average, DGGI improved the MMD (84.4 $\pm$ 27.3)%, (95.1 $\pm$ 8.9)%, (97.9 $\pm$ 3.5)%, and (94.7 $\pm$ 9.5)% for assortativity, betweenness, clustering, and node degree, respectively. Compared to training samples, the graphs generated by DGGI exhibit lower realism levels than those sampled from the training set, as expected, since the training set is drawn from the genuine population of intra-AS graphs (IGraphs). However, despite this disparity, DGGI demonstrates superior realism compared to baselines. Consequently, DGGI exhibits the most notable overall reduction in discrepancy concerning the MMD values.

VII. CONCLUSION

This paper introduced DGGI, a novel intra-AS graph generator. It also introduces a novel dataset (IGraphs) composed of real intra-AS graphs. To create IGraphs, we proposed an adaptation of the multilevel algorithm in [35] (FRM) that is a parameterized algorithm for subgraph extraction, which ensures that subgraphs are within predefined limits for a number of nodes without losing the original characteristics of the graph formation process.

The experimental results demonstrate that the generator DGGI outperforms all the baseline generators. On average, DGGI improved the Maximum Mean Discrepancy (84.4$\pm$ 27.3)%, (95.1$\pm$ 8.9)%, (97.9$\pm$ 3.5)%, and (94.7$\pm$ 9.5)%, for assortativity, betweenness, clustering, and node degree, respectively, as shown in Table III.

IGraphs is the first dataset to show such a wide variety of real-world intra-AS graphs, offering novel possibilities for the analysis of solutions for the Internet. IGraphs provides the possibility of training data-driven models based on graphs

using only real-world topologies and improving the generalization capacity of models due to the variety of graphs. IGraphs can also be used to diversify the simulation and emulation of solutions for the Internet.

Future work includes investigating DGGI's generalization capacity and using it for graph-based learning algorithms [3], [5]. We also plan to develop a user interface for DGGI to help synthesize graphs for intra-AS networks.

REFERENCES

[1] C. Labovitz, A. Ahuja, R. Wattenhofer, and S. Venkatachary, "The impact of Internet policy and topology on delayed routing convergence," in *Proc. Conf. Comput. Commun. 20th Annu. Joint Conf. IEEE Comput. Commun. Soc.*, 2001, pp. 537–546.

[2] M. H. Gunes and K. Sarac, "Importance of IP alias resolution in sampling Internet topologies," in *Proc. IEEE Glob. Internet Symp.*, 2007, pp. 19–24.

[3] J. Suárez-Varela et al., "Graph neural networks for communication networks: Context, use cases and opportunities," *IEEE Netw.*, vol. 37, no. 3, pp. 146–153, May/Jun. 2023.

[4] W. Jiang, "Graph-based deep learning for communication networks: A survey," *Comput. Commun.*, vol. 185, pp. 40–54, Mar. 2022.

[5] F. Tang, B. Mao, Y. Kawamoto, and N. Kato, "Survey on machine learning for intelligent end-to-end communication toward 6G: From network access, routing to traffic control and streaming adaption," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1578–1598, 3rd Quart., 2021.

[6] A. Vázquez, R. Pastor-Satorras, and A. Vespignani, "Large-scale topological and dynamical properties of the Internet," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 65, no. 6, 2002, Art. no. 066130.

[7] P. Mahadevan, C. Hubble, D. Krioukov, B. Huffaker, and A. Vahdat, "Orbis: Rescaling degree correlations to generate annotated Internet topologies," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2007, pp. 325–336.

[8] G. Accongiagioco, E. Gregori, and L. Lenzini, "S-BITE: A structure-based Internet topology generator," *Comput. Netw.*, vol. 77, pp. 73–89, Feb. 2015.

[9] B. Jiao and W. Zhang, "Structural decomposition model for the evolution of AS-level internet topologies," *IEEE Access*, vol. 8, pp. 175277–175296, 2020.

[10] K. Bakhshaliyev and M. H. Gunes, "Generation of 2-mode scale-free graphs for link-level Internet topology modeling," *PLoS ONE*, vol. 15, no. 11, 2020, Art. no. e0240100.

[11] S.-H. Yook, H. Jeong, and A.-L. Barabási, "Modeling the Internet's large-scale topology," *Proc. Nat. Acad. Sci.*, vol. 99, no. 21, pp. 13382–13386, 2002.

[12] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.

[13] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: An approach to universal topology generation," in *Proc. 9th Int. Symp. Model., Anal. Simul. Comput. Telecommun. Syst.*, 2001, pp. 346–353.

[14] N. Moshiri, "The dual-Barab\'asi-Albert model," 2018, *arXiv:1810.10538*.

[15] G. Bianconi and A.-L. Barabási, "Competition and multiscaling in evolving networks," *Europhys. Lett.*, vol. 54, no. 4, p. 436, 2001.

[16] R. Albert and A.-L. Barabási, "Topology of evolving networks: Local events and universality," *Phys. Rev. Lett.*, vol. 85, no. 24, p. 5234, 2000.

[17] T. Lappas, K. Pelechrinis, M. Faloutsos, and S. V. Krishnamurthy, "A simple conceptual generator for the internet graph," in *Proc. 17th IEEE Workshop Local Metrop. Area Netw. (LANMAN)*, 2010, pp. 1–6.

[18] A. Elmokashfi, A. Kvalbein, and C. Dovrolis, "On the scalability of BGP: The role of topology growth," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 8, pp. 1250–1261, 2010.

[19] C. L. Hood and G. F. Riley, "On predicting the performance characteristics of the NS-3 distributed simulator for scale-free Internet models," in *Proc. Workshop NS-3*, 2015, pp. 54–59.

[20] P. Barford, N. Duffield, A. Ron, and J. Sommers, "Network performance anomaly detection and localization," in *Proc. IEEE INFOCOM*, 2009, pp. 1377–1385.

[21] B. P. Swenson and G. F. Riley, "Simulating large topologies in ns-3 using BRITE and CUDA driven global routing," in *Proc. 6th Int. ICST Conf. Simul. Tools Techn.*, 2013, pp. 159–166.

[22] M. A. Canbaz, K. Bakhshaliyev, and M. H. Gunes, "Router-level topologies of autonomous systems," in *Proc. Int. Workshop Complex Netw.*, 2018, pp. 243–257.

[23] M. R. Mendonça, A. M. Barreto, and A. Ziviani, "Approximating network centrality measures using node embedding and machine learning," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 220–230, Jan.–Mar. 2021.

[24] Y. Feng, H. Wang, C. Chang, and H. Lu, "Intrinsic correlation with betweenness centrality and distribution of shortest paths," *Mathematics*, vol. 10, no. 14, p. 2521, 2022.

[25] M. Latapy and C. Magnien, "Complex network measurements: Estimating the relevance of observed properties," in *Proc. 27th Conf. Comput. Commun.*, 2008, pp. 1660–1668.

[26] R. Albert, H. Jeong, and A.-L. Barabási, "Error and attack tolerance of complex networks," *Nature*, vol. 406, no. 6794, pp. 378–382, 2000.

[27] L. Maccari and R. L. Cigno, "Pop-routing: Centrality-based tuning of control messages for faster route convergence," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, 2016, pp. 1–9.

[28] M. H. Gunes, S. Bilir, K. Sarac, and T. Korkmaz, "A measurement study on overhead distribution of value-added internet services," *Comput. Netw.*, vol. 51, no. 14, pp. 4153–4173, 2007.

[29] N. Hidaka, S. Arakawa, and M. Murata, "A modeling method for ISP topologies based on network-cost optimization," in *Proc. 4th Int. Conf. Autonom. Auton. Syst. (ICAS)*, 2008, pp. 169–174.

[30] C. Dadauto, N. Saldanha da Fonseca, and R. Silva Torres, "Internet graphs (IGraphs)." 2023. [Online]. Available: https://dx.doi.org/10.21227/cnw0-ea27

[31] "The CAIDA macroscopic Internet topology data kit." Aug. 2020. [Online]. Available: https://www.caida.org/catalog/datasets/internet-topology-data-kit

[32] A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola, "A Kernel method for the two-sample-problem," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19, 2006, pp. 513Ű-520.

[33] J. You, R. Ying, X. Ren, W. Hamilton, and J. Leskovec, "GraphRNN: Generating realistic graphs with deep auto-regressive models," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5708–5717.

[34] L. d. F. Costa, F. A. Rodrigues, G. Travieso, and P. R. Villas Boas, "Characterization of complex networks: A survey of measurements," *Adv. Phys.*, vol. 56, no. 1, pp. 167–242, 2007.

[35] L. Guillaume, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 10, Oct. 2008, Art. no. P1008. [Online]. Available: https://cir.nii.ac.jp/crid/1572824500601496448

[36] M. Simonovsky and N. Komodakis, "GraphVAE: Towards generation of small graphs using variational autoencoders," in *Proc. 27th Int. Conf. Artif. Neural Netw.*, 2018, pp. 412–422.

[37] Y. Li, O. Vinyals, C. Dyer, R. Pascanu, and P. Battaglia, "Learning deep generative models of graphs," 2018, *arXiv:1803.03324*.

[38] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proc. 8th Workshop Syntax, Semant. Struct. Statist. Transl.*, 2014, pp. 103–111.

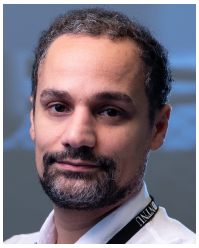
[39] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, Oct. 2011.

[40] F. Topsøe, "Bounds for entropy and divergence for distributions over a two-element set," *J. Ineq. Pure Appl. Math*, vol. 2, no. 2, p. 300, 2001.

[41] D. P. Kingma, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–15.

Caio Vinicius Dadauto received the B.Sc. degree in physics from the University of São Paulo in 2016, and the M.Sc. degree in applied mathematics from the University of Campinas in 2018, where he is currently pursuing the Ph.D. degree. His research interests include data science focusing on graph theory applied to communication networks.

Nelson L. S. da Fonseca (Senior Member, IEEE) received the Ph.D. degree in computer engineering from the University of Southern California in 1994. He is a Full Professor with the Institute of Computing, State University of Campinas, Campinas, Brazil. He has published 450+ papers and supervised 90+ graduate students. He is the recipient of the 2012 IEEE Communications Society (ComSoc) Joseph LoCicero Award for Exemplary Service to Publications, the Medal of the Chancellor of the University of Pisa in 2007, the Elsevier Computer Networks Journal Editor of Year 2001 Award, the Medal of the Chancellor of the University of Pisa in 2007, and the Elsevier Computer Network Journal Editor of the Year 2001. He is a former Editor-in-Chief of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS. He served as a ComSoc VP Conferences, a VP Technical and Educational Activities, a VP Publications, and a VP Member Relations, the Director of Conference Development, the Director of Latin America Region, and the Director of On-Line Services. He is currently an IEEE ComSoc Strategic Planning Committee Chair.



Ricardo Da Silva Torres (Member, IEEE) received the B.Sc. degree in computer engineering and the Ph.D. degree in computer science from the University of Campinas, Brazil, in 2000 and 2004, respectively. He was a Professor with the University of Campinas from 2005 to 2019. He has been a Professor of Visual Computing with the Norwegian University of Science and Technology since 2019. He is currently a Professor of Data Science and Artificial Intelligence with Wageningen University and Research. He has been developing multidisciplinary e-science research projects involving multimedia analysis, multimedia retrieval, machine learning, databases, information visualization, and digital libraries.