Triad Census in Large Directed Random Graphs given Degree Sequence

Kun Tu, Ananthram Swami, Don Towsley

Abstract

Triad census (countings of three-node subgraphs) provides a useful measurement to characterize the topological structure of a directed network. On the other hand, random graphs with the same degree distribution are useful to discover properties of a empirical network. Recent analysis on triad in both empirical and random directed graphs helps to identify network types (e.g., email or voting network) in high accuracy given graph topology. This results in useful real-world applications such as detecting relation or the way of interactions between individuals. However, it is time-consuming for current simulation based method to compute triad census in large random directed graphs given an in/out degree (or bi-degree) sequence. In this paper, we propose a fast approximation algorithm for computing average triad counting in large random graph given bi-degree sequence (BDS). Our algorithm, based on a configuration model, has a decreasing error with the increase on network size when the second moment of in/out degrees are finite. Experiments show that our algorithm can be applied to on a wide range of real-world networks. Compare to simulation based methods, our methods has up to 10X speed improvement and the error rates are kept below 5% for networks with size large than 1000. In the application of computing subgraph ratio profile (SRP) to cluster networks, the result from our method is identical to the simulation based method, suggesting our algorithm preferable for many large real-world networks.

1 Introduction

A network motif is a small subgraphs with a specific topology structure that repeatedly appears in a network. Triads are three-node motifs in a directed network. They are popular in network research because they provide more structural information than two-node motifs (dyads) and require less computational complexity than motifs with more than four nodes (which may even be computationally intractable in large networks). Many studies have shown that triad census, obtained by computing the frequencies of all types of motifs, enables us to distinguish different types of networks [17, 20, 12].

Analysis in triad census is usually performed using null models: random graphs with the same property (e.g., number of nodes and edges) as the network in question are generated in simulation. Then their average motif frequencies are computed and compare to those of the network to verify if the network contains some specific structural features. In the early study, a lot of work made the assumption that the probability of an edge between two nodes is i.i.d. and used that probability to generate random graphs [13]. With this assumption, the probability can be approximate using Poisson distribution in the limit of large graph size [19] and thus provide convenient way to study features of random graphs. However, recent studies [2, 19] have discovered that a wide range of real-world networks, such as food webs[11], social networks [24], world-wide web [3] and biological system networks [10], have distributions of node degrees that are different from Poisson distribution. As a result, it is suggested [19] that the ordinary (Poisson) random graph is not an adequate framework to uncover important features of these real-world networks. Current study on triad census with null models has been focusing on random graphs with the same degree sequence or distribution as the network of interest [17].

In this paper, we focus on computing triad census for random directed graphs given in/out degree sequence (also called bi-degree sequence or BDS). Related works includes two different approaches, a simulation based method and an analytical model.

In simulation, an algorithm generates sufficient amount of random graphs for the purpose of statistic significance and then compute motif census in those graphs. Fast algorithms have been proposed for exact motif census in a small network [1] with time complexity $O(ND^2)$, where N is number of nodes and D is

the average (sum of in/out) degree of a node. For a large scale network, sampling techniques are applied to approximate the motif census [23]. On the other hand, it is most computational challenging to uniformly generate large random graphs with the constraint of BDS. Methods for directed random graph generation are derived from those for undirected graphs. One way [16] is to generate a random adjacency matrix by sampling edges using degree distribution constructed from the BDS. In practice, however, this method only works well with small networks because its large computational complexity both in space and time. An alternative method iteratively generates directed edges and restart the process from scratch if generation of self-loops or multi-edges are unavoidable. It improves on the space complexity but the computation is intractable in large networks with no upper bound of the time complexity due to the restart of the graph generation process. Several methods [17, 8] first constructs a simple directed graph using methods derived from Havel-Hakimi algorithm, then use Markov-chain methods to rewire edges to generate new random graphs. The rewiring process from these methods improve on the computational speed of the previous method but fails to guarantee a uniform sampling of the random graphs because some graph structures may not be obtainable from a pre-constructed graph [8]. Moreover, algorithms for these methods are still too slow for large networks because extra operations are required to avoid self-loops and multi-edges during the rewiring. To construct large networks, researchers propose configuration models [7]: A node u with in degree I_u and out degree O_u is considered to have I_u "in-stub" and O_u "out-stub" for edges; a directed edge is generated by randomly choosing an out-stub and an in-stub of nodes. This approach is fast with time complexity $O(|E|)$, where $|E|$ is the number of edges. However, a random graph generated from this method is not a simple graph. Although the self-loops and multi-edges can be removed to construct a simple graph, the BDS is distorted and causes errors when computing triad census. In practice, the methods above are chosen based on the requirement of the accuracy and the speed for computing the triad census. Note that the required number of random graphs increases the time complexity to compute average triad census (with sample size at least 4899 when confidence level is 95% and confidence interval is $\pm 1\%$). As a result, it is usually time consuming to use simulation methods to compute motif census in large graphs.

Analytical models, on the other hand, avoid the sampling process by implementing a probability model to compute the expected triad census and improve the computational. Approximated probabilities of triads given bi-degree pairs of three nodes are computed based on a configuration model that allows self-loops and multi-edges. If the second moment of in/out degree is finite, the expected number of self-loops and multi-edges remains constant as the network size grows, resulting in trivial error that can be ignored in large networks. There are many methods for different null models [9], but only a few are proposed for random graphs with BDS: Algorithms based on minimum description length [4] detect motifs in undirected network but difficult to extend to directed graphs. The most related method [22] implements maximum likelihood estimation of random graph [21] to detect triads in world trade network. However, this model ignores the constraints from the node degrees in the triad and have large error in some situations. Moreover, the algorithm for this model requires $O(N^6)$ time complexity and is slow in large networks, where N is the size of a network.

Based on previous study, we introduce a probability model approximate the expected frequency of triads in large directed random graphs with BDS and a fast algorithm to compute the triad census. Our model is also based on configuration model but provide a mechanism to reduce the error caused by self-loop and multi-edges. Our algorithm is faster than current existing mode, with time complexity of $O(\tilde{D}^3)$, where $\tilde{D}$ is the number of unique in/out degree pairs in the network. Experiments show that our model is faster than simulation and analytical methods and has more accurate result than current analytical models. The improvement on speed from our model is even more significant when the network size is large and $\tilde{D}$ is finite.

The rest of the paper is organized as follow: we first provide basic concepts for motifs and configuration model (Section 2). Then we introduce our model that computes triad frequencies based on the probabilities of edges between nodes given a BDS (Section 3) and explain the how to approximate the expected triad census using configuration model (Section 4). We also provide a fast algorithm and analyze its time complexity (Section 5). Finally, we evaluation our model applying it to synthetic and multiple real-world dataset and compare the result to other simulation and analytical methods (Section 6).

2 Preliminary

2.1 Network Motifs in Directed Graphs

A dyad motif is a subgraph containing only two nodes. Given a directed graph $G(V, E)$, where V is a set of nodes and E is a set of directed edges, let $u, v \in V$ be nodes in a dyad, $e_{u,v} \in V$ is a directed edge in E, there are four possible topological structures for a subgraph consisting u, v (Table 1). These four structures are classified as three isomorphic dyad classes: null dyad (with no edges), asymmetric dyad (with only one directed edge) and mutual dyad (contain reciprocal edges). Dyad motifs are basic structures to study other

Table 1: Four Possible Structures for node u, v and their edges

Dyad Type	1	2	3	4
Directed Edges	N/A	$e_{u,v}$	$e_{v,u}$	$e_{u,v}, e_{v,u}$

motifs because properties of subgraphs can be derived from dyads.

A triad motif is a subgraph containing three nodes that connects one another. There are totally 16 triad isomorphism classes based on the topological structure in a directed graph(Fig 1).

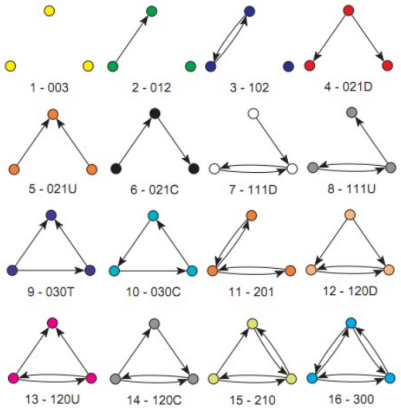


Figure 1: Triad Isomorphism Classes (MAN labeling) [**?**]. An MAN label consists of three digits and an optional character. The first digit represents the number of mutual connections between two nodes (E.g., the 16th triad, labeled with "300", has three mutuals). The second digit is the number of asymmetric connections (e.g., the 9th triad labeled with "030T"). The last digit denotes the number of null edges (e.g., the 11th triad, labeled with "201", contains a pair of nodes with no connection). The optional character represents the directions of edges: "D" for down; "U" for up; "T" for transition and "C" for cyclic.

2.2 Configuration Model for Directed Graph with In/out Degree Sequence

It is difficult to uniformly generate a simple random graph given a degree sequence because the edge generation with the constraint imposed by the bi-sequence is usually computational intractable. A configuration model is first proposed for fast realization of undirected random graphs with self-loop and multi-edges given a degree sequence. It is pointed out [18] that the density of self-loops and multi-edges tend to decrease as network size increases and results in a good approximation of large simple graphs. It can be easily extended to directed graphs with bi-sequence: a node u with in degree I_u and out degree O_u is considered to have I_u "in-stubs" and O_u "out-stubs" for directed edges. A edge is generated by randomly choosing an in-stub and an out-stub from all the nodes.

It is proven that [5] for a network with size N, the expected number of self-loops, denoted by S_N, or multi-edges, denoted by M_N, approaches to a non-zero mean of Possion distribution as the size of the network

increase to infinity:

$$\lim_{N\to\infty} S_N = \frac{E[I_u O_u]}{\mu} = \frac{< IO >}{\mu} \quad (1)$$

$$\lim_{N\to\infty} M_N = \frac{E[I_u(I_u-1)O_u(O_u-1)]}{2\mu^2} = \frac{(< I^2 > - < I >)(< O^2 > - < O >)}{2\mu^2} \quad (2)$$

where $< X^m >= \frac{1}{N}\sum_{u=1}^{N} X_u^m$ represents the m-th moment of variable X, and $\mu = E(I_u) = E(O_u) > 0$. As long as $< IO >, < I^2 >$ and $< O^2 >$ are finite, S_N and M_N is a constant. This implies that S_N and M_N are vanishing $O(1/N)$ fractions of total edges in large-N limit. That is, the probability of a node with a self-loop or multi-edges is small in large networks. This enable us to design a fast algorithm to approximate triad properties of large simple graphs using edge probabilities derived from configuration models. In the following sections, we consider the constraints from the bi-sequence and node connectivities in triads and use edge probability based on configuration models to approximate the expected triad frequencies with better accuracy.

3 Probability Model for Triad Census in Random Graph of Bi-Degree Sequence

Let G be a random directed graph of N nodes with an in-degree sequence, denoted by $\vec{I} = [I_1, I_2, \ldots, I_N]$, and an out-degree sequence, denoted by $\vec{O} = [O_1, \ldots, O_N]$, we want to compute the expected frequencies of triads by estimating the distribution of triads formed by any three nodes (u, v and w,) given their in/out degrees tuple, denoted by $D(u, v, w) = (I_u, O_u, I_v, O_v, I_w, O_w)$.

We define $\mathcal{T}_{u,v,w} \in \{1, 2, \ldots, 16\}$ as the type of triad formed by node u, v and w(Fig.1) and $\mathcal{D}_{u,v} \in \{1, 2, 3, 4\}$ as the type of dyad formed by u and v (Table 1). The probability of triad type of a three-node subgraph given the in/out degrees is $P(\mathcal{T}_{u,v,w}|D(u, v, w), \vec{I}, \vec{O})$. To simplify notations, we use $P(\mathcal{T}_{u,v,w})$ to denote $P(\mathcal{T}_{u,v,w}|D(u, v, w), \vec{I}, \vec{O})$. Similarly, we denote $P(\mathcal{D}_{u,v}|I_u, O_u, I_v, O_v, \vec{I}, \vec{O})$ using $P(\mathcal{D}_{u,v})$, $P(\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}|D(u, v, w), \vec{I}, \vec{O})$ using $P(\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w})$ in the rest of the paper.

Since a triad $\mathcal{T}_{u,v,w}$ is determined by the three dyads $\mathcal{D}_{u,v}, \mathcal{D}_{u,w}$ and $\mathcal{D}_{v,w}$, we can compute the probability that u, v and w forms a specific triad as follow:

$$P(\mathcal{T}_{u,v,w}) = \sum_{\mathcal{D}_{u,v},\mathcal{D}_{u,w},\mathcal{D}_{v,w}} P(\mathcal{T}_{u,v,w}|\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}) P(\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}) \quad (3)$$

where $\mathcal{T}_{u,v,w}$ is independent of the in/out degrees once the dyads $\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}$ are fixed. $P(\mathcal{T}_{u,v,w}|\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}) \in \{0, 1\}$ can be easily determined based on whether a triad constructed by the three dyads $\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}$, matches the triad type $\mathcal{T}_{u,v,w}$.

Finally, the expected frequency of triad t is the sum of probability of t from all possible three-node subgraphs:

$$\sum_{(u,v,w)} P(\mathcal{T}_{u,v,w} = t) \quad (4)$$

3.1 Basic Probability Model for Isomorphic Three-node Subgraph

There are totally 64 different structures for a three-node subgraph, each determined by the dyad tuple $(\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w})$.

Some subgraph structures are isomorphic and belong to the same triad type. For example, The fourth triad "021D" in Fig. 1 has three isomorphic structures determined by dyad tuples $(\mathcal{D}_{u,v} = 2, \mathcal{D}_{u,w} = 2, \mathcal{D}_{v,w} = 1)$, $(\mathcal{D}_{u,v} = 1, \mathcal{D}_{u,w} = 2, \mathcal{D}_{v,w} = 2)$ and $(\mathcal{D}_{u,v} = 2, \mathcal{D}_{u,w} = 1, \mathcal{D}_{v,w} = 2)$, respectively. For the three structures, we set $P(\mathcal{T}_{u,v,w} = 4|\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) = 1$. Otherwise $P(\mathcal{T}_{u,v,w} = 4|\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) = 0$. More generally, we define a function of $\mathcal{D}_{u,v}, \mathcal{D}_{v,w}$ and $\mathcal{D}_{u,w}$ that maps a three-node subgraph structures to a triad type, denoted as

$$g(\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) = j, j \in [1, \ldots, 16]$$

Computing the value of $g(\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w})$ is trivial because a triad type can be easily determined by the dyads. Thus, we have:

$$P(\mathcal{T}_{u,v,w} = g(\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) | \mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) = 1$$
$$P(\mathcal{T}_{u,v,w} \neq g(\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) | \mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) = 0$$

We can compute the probability of the t-th triad according to Eq(3):

$$P(\mathcal{T}_{u,v,w} = t) = \sum_{g(\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}) = t} P(\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w}) \tag{5}$$

The expected frequency of triad t is the sum of probability of t from all possible three-node subgraphs:

$$\sum_{(u,v,w \in V)} P(\mathcal{T}_{u,v,w} = t) = \sum_{(u,v,w \in V)} \sum_{g(\mathcal{D}_{uv}, \mathcal{D}_{uw}, \mathcal{D}_{vw}) = t} P(\mathcal{D}_{uv}, \mathcal{D}_{vw}, \mathcal{D}_{uw}) \tag{6}$$

where V is the set of nodes in network G.

3.2 A Fast Approximation Model and Notation

In Eq(6), we need to compute triad probability for all possible $\frac{N(N-1)(N-2)}{6}$ combinations of three-node subgraphs, resulting in an $O(N^3)$ time complexity for computation. In practice, some nodes have the same in/out degrees, as a result, some subgraphs may have the same degree tuple $D(u, v, w)$. We modify the basic probability model and improve the time complexity by computing every unique in/out degree subgraph once.

In the rest of the paper, we re-define some notations to distinguish the modified model from the basic model in Section 3.1 (Table 2). We call the in/out degree pair of a node as bi-pair. u, v and w represent the bi-pairs instead of nodes, N_u represents the number of nodes with in/out degree that equal to bi-pair (I_u, O_u). We define $S = \{(I_u, O_u)\}$ as a set of distinct bi-pairs. $\mathcal{D}_{u,v}$ represents a dyad type formed by two nodes with bi-pair u, v and $\mathcal{T}_{D(u,v,w)}$ is the triad type formed by nodes with bi-pair u, v and w.

Table 2: Re-Definition of Notation for the Fast Model

Notation	Meaning
S	a set of unique bi-pairs $\{(I_i, O_i)\}$ in graph G
$u \in S$	an bi-pairs (I_u, O_u)
$\mathcal{D}_{u,v}$	a dyad type formed with $u, v \in S$
$\mathcal{T}_{u,v,w}$	a triad type formed with $u, v, w \in S$
$D(u, v, w)$	bi-pairs in a triad $(I_u, O_u, I_v, O_v, I_w, O_w)$

Since $D(u, v, w)$ decide $P(\mathcal{T}_{u,v,w} = t)$, Eq(5) can be re-written as the probability of a triad type t given bi-pair u, v, w:

$$\begin{aligned} P(\mathcal{T}_{u,v,w} = t) &= \sum_{g(\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}) = t} P(\mathcal{D}_{u,v}, \mathcal{D}_{v,w}, \mathcal{D}_{u,w} | u, v, w) \\ &= \sum_{g(\mathcal{D}_{u,v}, \mathcal{D}_{u,w}, \mathcal{D}_{v,w}) = t} P(\mathcal{D}_{u,v}) P(\mathcal{D}_{u,w} | \mathcal{D}_{u,v}) P(\mathcal{D}_{v,w} | \mathcal{D}_{u,w}, \mathcal{D}_{u,v}) \end{aligned} \tag{7}$$

where $P(\mathcal{D}_{u,v}), P(\mathcal{D}_{u,w} | \mathcal{D}_{u,v})$ and $P(\mathcal{D}_{v,w} | \mathcal{D}_{u,w}, \mathcal{D}_{u,v})$ can be approximated using edge probabilities based on a configuration model (Section 4).

Let $c(u, v, w)$ be the number of three-node subgraphs of unique combination of bi-pairs u, v and w, the frequency of triad t (Eq(6)) can be computed as:

$$\sum_{(u,v,w \in S)} P(\mathcal{T}_{u,v,w} = t) = \sum_{D(u,v,w)} c(u, v, w) P(\mathcal{T}_{u,v,w} = t) \tag{8}$$

where the computaiton of $c(u, v, w)$ is provided in Appendix B. The time complexity is determined by the number of unique degree tuples of $D(u, v, w)$, which is much smaller than $O(N^3)$ of the basic model in practice. (We will explain the time complexity in details in Section **??**.)

4 Triad Census Approximation

Since the connection between nodes should satisfies the constraint by the in/out degree sequences. The computation of the probability of a dyad should be conditional on other dyads and is intractable in large networks. To address this issue, we apply a configuration model for directed graph to approximate the triad distribution of a large graph given the BDS by assuming independence among triads.

4.1 Probability of Node Connection in Configuration Model

Let $G(V, E)$ be a random directed graph with V as a set of nodes and E as a set of directed edges, $[(I_1, O_1), \ldots, (I_{|V|}, O_{|V|})]$ is the in/out degree sequence for the nodes, The number of edge $|E| = \sum_{u=0}^{|V|} O_u$. A configuration model generates a directed edge, denoted as $e(u, v)$, by wiring one of the O_u stubs from u to one of the I_v stubs from v. The configuration model allows self-loop and multi-edges in a graph. However, if $I_u \ll |E|, O_u \ll |E|$ for $u = 1, \ldots, |V|$, the number of self-loop or multi-edges are proven to converge to a constant given a fixed joint in/out degree distribution [6]. As the number of nodes/edges in a graph increase and the upper bound of in/out degree is much smaller than the graph size, the effect of self-loop or multi-edges on the probability of an edge is small.

We consider a directed edge goes from an out-stub of a node to an in-stub of the other node. Suppose we only allow multi-edges in G, the probability that an out-stub of u connects to an in-stub of v is $\frac{I_v}{|E|-I_u}$. Then, the probability that there are no directed edges from u to v is

$$\bar{p}_{uv} = \prod_{j=0}^{O_u-1} (1 - \frac{I_v}{|E| - I_u - j}) \tag{9}$$

When $O_u \ll |E| - I_u$, we have $|E| - I_u \approx |E| - I_u - j$ for $j = 0, \ldots, O_u - 1$, Eq(9) can be approximated by

$$\bar{p}_{uv} \approx (1 - \frac{I_v}{|E| - I_u})^{O_u} \tag{10}$$

If $|E| - I_u$ is much larger than $O_u I_v$, first-order approximation can be applied for the right side of Eq(10) to speed up computation:

$$\begin{aligned} \bar{p}_{uv} &\approx 1 - \frac{O_u I_v}{|E| - I_u} + \ldots \\ &\approx 1 - \frac{O_u I_v}{|E| - I_u} \end{aligned} \tag{11}$$

As a result, we can approximate the probability that u connects to v in a directed random graph with no self-loop:

$$p_{uv} \approx \frac{O_u I_v}{|E| - I_u} \tag{12}$$

Finally, if $I_u \ll |E|$, we have $|E| \approx |E| - I_u$. The probability that there is at least one directed edge from u to v can be approximated as:

$$p_{uv} = \frac{O_u I_v}{|E|} \tag{13}$$

In practice there is situation when $\frac{O_u I_v}{|E|} > 1$, we set

$$p_{uv} = \min(1, \frac{O_u I_v}{|E|}) \tag{14}$$

4.2 Probability of Dyads

The probability of a dyad according to Eq(9) is

$$P(\mathcal{D}_{u,v}=i)\approx\begin{cases}\prod_{j=0}^{O_u-1}(1-\frac{I_v}{|E|-I_u-j})\prod_{j=0}^{O_v-1}(1-\frac{I_u}{|E|-I_v-j}) & \text{if } i=1\\ (1-\prod_{j=0}^{O_u-1}(1-\frac{I_v}{|E|-I_u-j}))\prod_{j=0}^{O_v-1}(1-\frac{I_u}{|E|-I_v-j}) & \text{if } i=2\\ \prod_{j=0}^{O_u-1}(1-\frac{I_v}{|E|-I_u-j})(1-\prod_{j=0}^{O_v-1}(1-\frac{I_u}{|E|-I_v-j}) & \text{if } i=3\\ (1-\prod_{j=0}^{O_u-1}(1-\frac{I_v}{|E|-I_u-j}))(1-\prod_{j=0}^{O_v-1}(1-\frac{I_u}{|E|-I_v-j})) & \text{if } i=4\end{cases} \tag{15}$$

However, we can approximate it using Eq(13):

$$P(\mathcal{D}_{u,v}=i)\approx\begin{cases}(1-\frac{O_uI_v}{|E|})(1-\frac{O_vI_u}{|E|}) & \text{if } i=1\\ \frac{O_uI_v}{|E|}(1-\frac{O_vI_u}{|E|-1}) & \text{if } i=2\\ (1-\frac{O_uI_v}{|E|-1})\frac{O_vI_u}{|E|} & \text{if } i=3\\ \frac{O_uI_v\cdot O_vI_u}{|E|(|E|-1)} & \text{if } i=4\end{cases} \tag{16}$$

Note that $|E|(|E|-1)$ in the denominator can be approximated by $|E|^2$ when $O_uI_v \ll |E|, O_vI_u \ll |E|$.

Let E' be the set of directed edges, (I'_u, O'_u) be the in/out degree of u after $\mathcal{D}_{u,v}$ is decided. $O'_u = O_u - 1, I'_v = I_v - 1$ if there is an edge $e(u,v)$ from u to v. $P(\mathcal{D}_{u,w}|\mathcal{D}_{u,v})$ can be approximated by:

$$P(\mathcal{D}_{u,w}=i|\mathcal{D}_{u,v})\approx\begin{cases}\frac{(|E'|-O'_uI'_w)(|E'|-O'_wI'_u)}{|E'|^2} & \text{if } i=1\\ \frac{O'_uI'_w(|E'|-O'_wI'_u)}{|E'|(|E'|-1)} & \text{if } i=2\\ \frac{(|E'|-O'_uI'_w)O'_wI'_u}{|E'|(|E'|-1)} & \text{if } i=3\\ \frac{O'_uI'_w\cdot O'_wI'_u}{|E'|(|E'|-1)} & \text{if } i=4\end{cases} \tag{17}$$

Similarly, let E'' be the set of directed edges after $\mathcal{D}_{u,v}, \mathcal{D}_{u,w}$ are decided, we can approximate $P(\mathcal{D}_{v,w} = i|\mathcal{D}_{u,v}, \mathcal{D}_{u,w})$

$$P(\mathcal{D}_{v,w}=i|\mathcal{D}_{u,v},\mathcal{D}_{u,w})\approx\begin{cases}\frac{(|E''|-O''_uI''_w)(|E''|-O''_wI''_u)}{|E''|^2} & \text{if } i=1\\ \frac{O''_uI''_w(|E''|-O''_wI''_u)}{|E''|(|E''|-1)} & \text{if } i=2\\ \frac{(|E''|-O''_uI''_w)O''_wI''_u}{|E''|(|E''|-1)} & \text{if } i=3\\ \frac{O''_uI''_w\cdot O''_wI''_u}{|E''|(|E''|-1)} & \text{if } i=4\end{cases} \tag{18}$$

4.3 Triad Census in a Random Graph with Fixed In/Out Degree Sequences

For a directed random graph $G(VE)$, where V is a set of node and E is a set of directed edges, let $\vec{\mathcal{I}} = [I_1, I_2, \dots], \vec{\mathcal{O}} = [O_1, O_2, \dots]$ be the in/out degree sequences, $u, v, w \in V$ are three nodes in G. Given the tuple of in/out degree $D(u,v,w) = (I_u, O_u, I_v, O_v, I_w, O_w)$, the probability of triad type t, denoted as $P(\mathcal{T}_{D(u,v,w)} = t)$ in Eq(7), can be computed using Eq(16 - 18). (In Section A from the Appendix, we provide formula to computer the probabilities of 16 unique isomorphic three-node subgraphs in Fig.**??** given the in/out degree of the nodes.)

5 Algorithm and Time Complexity

With the approximation model in Section 4, we design a fast algorithm to compute Eq(8) to obtain triad frequencies given a bi-degree sequence (BDS) from a directed graph (Algrithm 1). It first obtain a set of unique bi-degree pairs from the BDS, denoted as S and compute the number of nodes that have bi-degree pair (I_u, O_u), denoted as N_u (Line 1-3). We apply Eq(25) and Eq(24) to obtain the expected frequencies for triads for every three degree pairs $D(u,v,w)$ (Line 4-8). Let $\tilde{D} = |S|$ be the number of unique bi-degree pairsthe time complexity is $O(\tilde{D}^3)$ according to Line 4-8.

Expected Error Using the expected number self-loops and mutli-edges in Eq(2), we compute the expected error in triad frequencies caused by a configuration model in a large N limit.

Algorithm 1: Average Triad Census for Graph given Bi-Degree Sequence (BDS)

Data: BDS $B = [(I_1, O_1), \ldots, (I_n, O_n)]$
Result: 16 avg triad frequences $[f_1, \ldots, f_{16}]$

```
S = getUniqueBiDegreePair(B) ;
for (I_u, O_u) ∈ S do
    p_u = (N_u, I_u, O_u) = getNumOfNodeWBiDegree(I_w, O_w)
for u = 1 : |S| do
    for v = u : |S| do
        for w = v : |S| do
            for t = 1 : 16 do
                f_t += getExpectedFreq(t, p_u, p_v, p_w)
```

First note that in a graph G of size N, if a directed edge between two node u, v is modified, totally $N-2$ triads containing u, v will change their triad type.

For a self-loop generated by a configuration model, at most five pairs of nodes have their edges changed (Fig 2), causing $5(N-2)$ triads to change their types. In large-N limit, the expected number of self-loop (S_N in Eq(2)) is considered as a constant, the fraction of number of mis-computed triad type is calculated as

$$\frac{5S_N(N-2)}{N(N-1)(N-2)} = \frac{5S_N}{N(N-1)} = O(\frac{1}{N^2}) \tag{19}$$

Figure 2: Two possible ways to rewire edges to generate a self-loop. (a)

Similarly, at most three pairs of nodes are required to rewired to generate a multi-edge (Fig 3) and the fraction of number of mis-computed traid type caused by multi-edges is calculated as

$$\frac{3M_N(N-2)}{N(N-1)(N-2)} = \frac{3M_N}{N(N-1)} = O(\frac{1}{N^2}) \tag{20}$$

Figure 3: Rewiring edges to generate a multi-edge

Both Eq (19) and Eq (20) show that although the error in counting numbers of triad types increases linearly with the network size, it is a vanishing $O(\frac{1}{N^2})$ fraction of the total triad number. In the Appendix E, we further show that the error in the number for each triad type is also vanishing with the increase in network size.

6 Experiments and Results

In this section, we evaluate the accuracy of motif census and time complexity of our model by applying it to both synthetic networks and real world networks given differernt bi-degree sequences (BDS) and comparing to simulation based methods.

6.1 Baseline Methods

We use two simulation based algorithms as baseline methods. The first one (Algorithm 3), widely used in current researches in null modle with given BDS [17], applies edge-rewiring method to generate simple random graphs and count the motif census for each graph. A second method (Algorithm 4) from iGraph

[7] is applied to generate graphs using configuration model to compute an approximated result. We also implement the analytical method [22]

Setting of simulation based method The number of random graphs generated in simulation, denoted by N, affects the estimate of average frequency a motif as well as the computational time. Several research [**?**, **?**] suggest that $N \geq 5000$ random graphs should be generated for large networks, which helps to obtain a good estimate of the motif census with 95% confidence level (CL) within 1% confidence interval (CI). However, $N = 5000$ results in large computational complexity of the baseline methods. To keep a good balance of the quality of estimation and computational speed, we choose $N = 1000$ in our experiments, which provide us with 95% CL and about 3% CI.

6.2 Synthetic data

We use iGraph to generate ER graphs, BA graphs and smallworld directed networks as synthetic data to test our method because these three types of networks have similar properties (such as network structures or degree distributions) as real-world networks. The sizes of the networks range from 100 to $10,000$. We set the average in/out degree within interval $[2, 10]$ (the same range in a fundamental report [15] and the real-world networks we test). For each setting of synthetic network, we generate 400 network instances.

6.2.1 Metrics

Accuracy We use result from Algorithm 3 as baseline for accuracy because the ground truth of average motif frequencies is computational intractable.

Let $\{G_i\}$ be a set of empirical networks, $\bar{f}_i$ be the average frequency of motif obtained Algorithm 3 for G_i's null model with the same BDS and $\hat{f}_i$ be the expected motif frequency from our model, we use a normalized mean absolute error (NMAE) to evaluate the accuracy of our model for different graphs:

$$\text{NMAE} = \frac{1}{n} \sum_{i=1}^{n} \frac{|\bar{f}_i - \hat{f}_i|}{\bar{f}_i + \epsilon} \tag{21}$$

where $\bar{f}_i + \epsilon$ is a normalization term so that we can compare error in our method accross graphs different in sizes and degrees, etc. The NMAE measures the error as a percentage of the motif frequency $\bar{f}_i$ in the baseline simulation based method. ϵ is a constant to deal with the situation when a motif type rarely appears ($\bar{f}_i < 1$), expecially when $\bar{f}_i = 0$. We set $\epsilon = 1$ so that when $P_i = 0$ we can use M_i to evalute the counting error. From statistic point of view, a NMAE below 5% of $\bar{f}_i$ is considered as acceptable.

Running Time For simulation based method, we compute the runtime by summing up the time to generate random graphs given a BDS and the time to compute motif census.

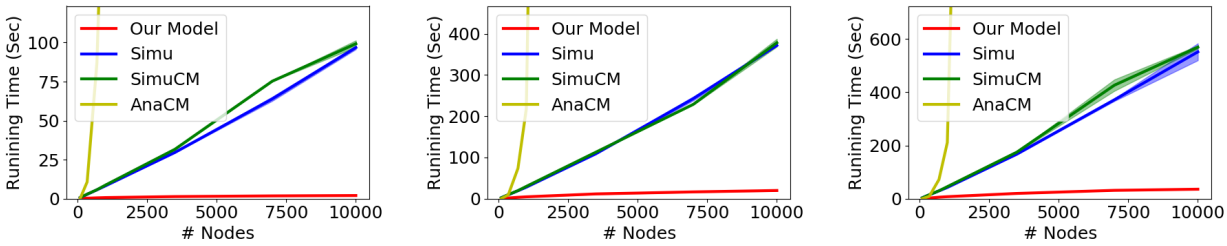
6.2.2 Results

We compare the time complexity of our method to the simulation based methods. Figure 4 shows the comparison of runtime for all methods to obtain average motif frequencies in networks of different sizes and average in/out degree.

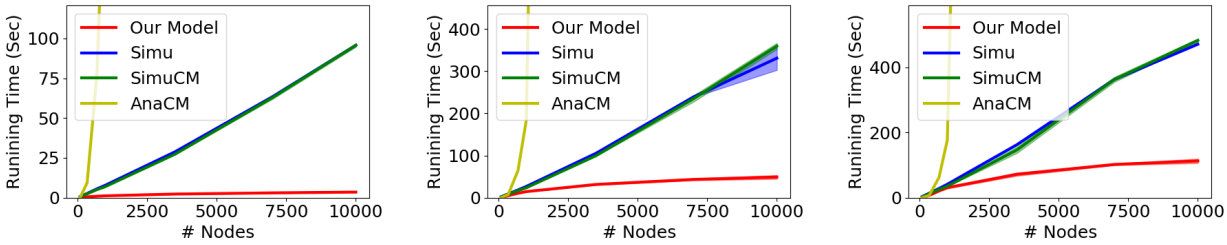
Figure (5 - 7) show the effect of network size on the NMAE of triad frequencies. For all 16 triads, the NMAEs for both our model and and simulation based method using configuration model (simuCM) decrease with the increase in network size. This is consistant with our analysis in Section 5. For ER graphs and small world networks, both methods reach an acceptable NMAE below 0.05 when the network size is larger than 400, suggesting they can be applied to large networks to obtain an approximation of average triad census from null model with a fixed BDS. Computing for BA graphs (Figure 7) is challenging to simuCM, it requires network size larger than 10^4 for simuCM to reach 0.05 NMAE for motif 9, 12 and 13. On the other hand, our method can be applied to networks with size larger than 10^3. Thus, the requirement of network size for our model is 10X smaller than simuCM. Moreover, the NMAEs of our method are significantly smaller than simuCM for all the random graphs.

6.3 Real-world Data and Application

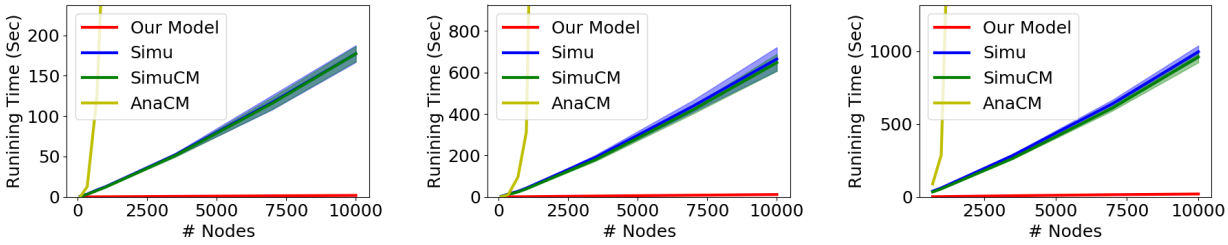
We next investigate if our method can apply to real-world networks of different types and sizes.



(a) ER Graphs with Avg Degree 4 (Left), 8 (Mid) and 10 (Right)



(b) Smallworld Network with Avg Degree 4 (Left), 8 (Mid) and 10 (Right)



(c) BA Graphs with Avg Degree 4 (Left), 8 (Mid) and 10 (Right)

Figure 4: Comparison on Running Time for Our Model and Two Simulation Based Methods to Obtain Average Motif Frequencies. X axis represents the number of nodes in a network. Y axis represent runtime (in seconds) to obtain averge frequencies for all motifs. Sub-figure (a) (b) and (c) shows results for ER graphs, smallworld networks and BA graphs, respectively. Solid lines represent the average runtimes and shaded areas represent the 5% - 95% percentiles. Performances in computational speed for both simulation based methods are similar and sensitive to the increase in number of nodes. Larger average in/out degree results in more computational time for all methods. Since the time complexity of our model only depends on the number of unique degree pairs and this number is relatively small compared to the size of the network, our model runs much faster than the simulation based methods for large networks.

6.3.1 Datasets

We use different directed networks from SNAP [14], such as social network (socBitcoin-otc), question-answering network (mathoverflow, askUbuntu) and communication networks (emailEU, emailEnron). Table 3 list the statistics of dataset properties that are related to time complexities of the tested algorithms.

6.3.2 Metric

In real-world application, triad 1 - triad 3 can be studied using dyads. As a result, the 13 triads whose nodes are connected (triad 4 - triad 16) receive more attention. To study the 13 triads across networks of different sizes and domains, their frequencies are usually normalized as a probability distribution. As a result, we use KL-divergence to evaluate the accuracy of our model across different datasets.

Let $P = [p_1, \ldots, p_n]$ be the distribution computed from baseline method and $Q = [q_i, \ldots, q_n]$ be the

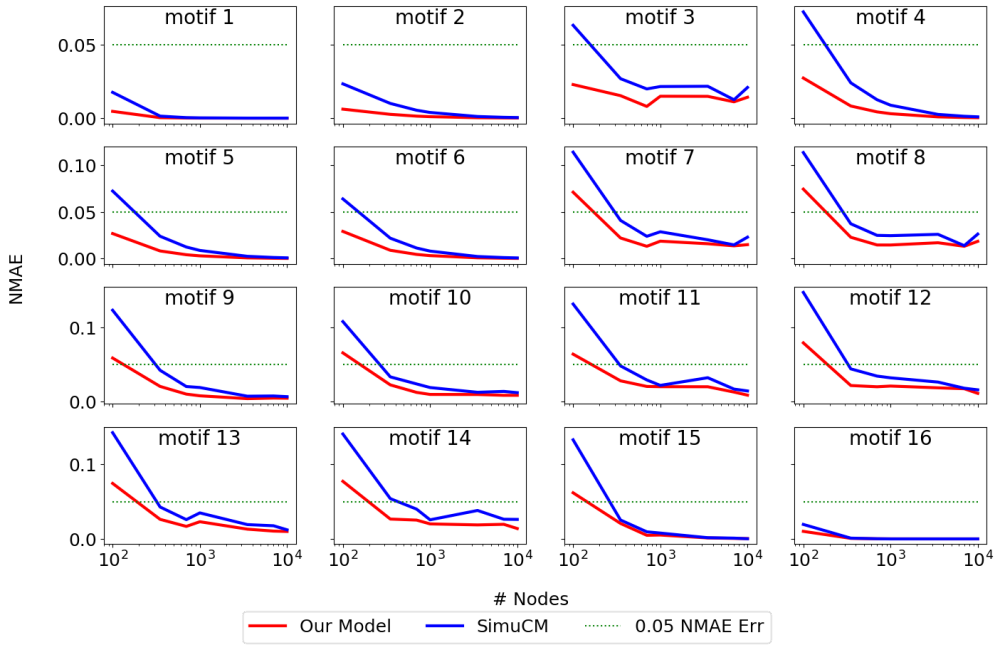


Figure 5: Network size and normalized mean absolute error (NMAE) of our model and simulCM for ER graphs.

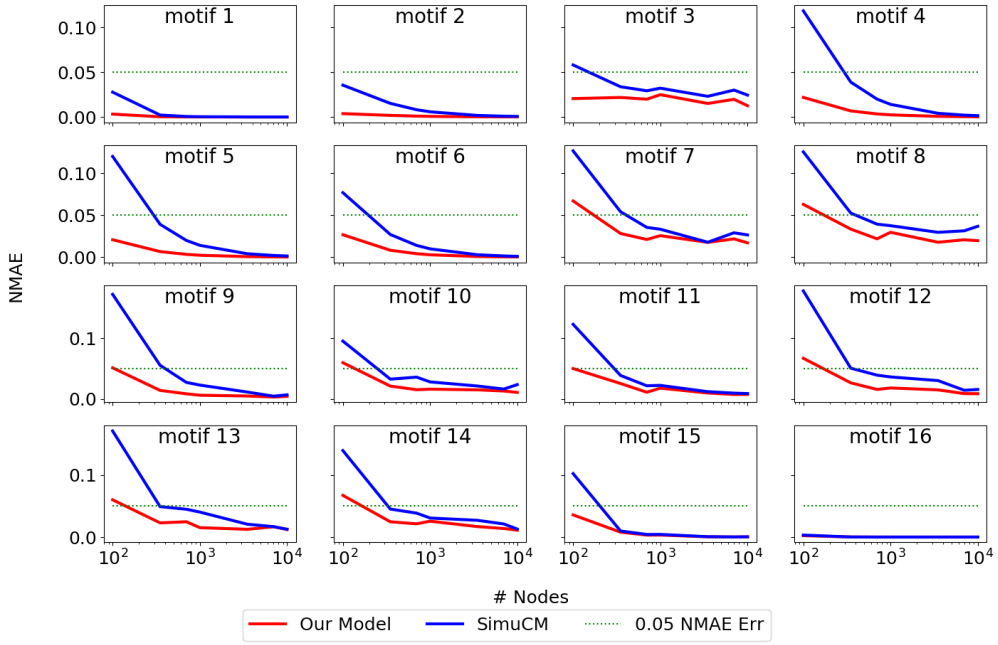


Figure 6: Network size and NMAE of our model and simuCM for small world graphs.

distribution from our method or the simulation based method with configuration model, the KL-divergence from Q to P is defined as

$$D_{KL}(P||Q) = \sum_i p_i \log \frac{p_i}{q_i} \geq 0.$$

$D_{KL}(P||Q) = 0$ indicates that $P = Q$. It provides a measure of how Q diverges from P, which enable us to investigate whether our method fit in different type of networks.

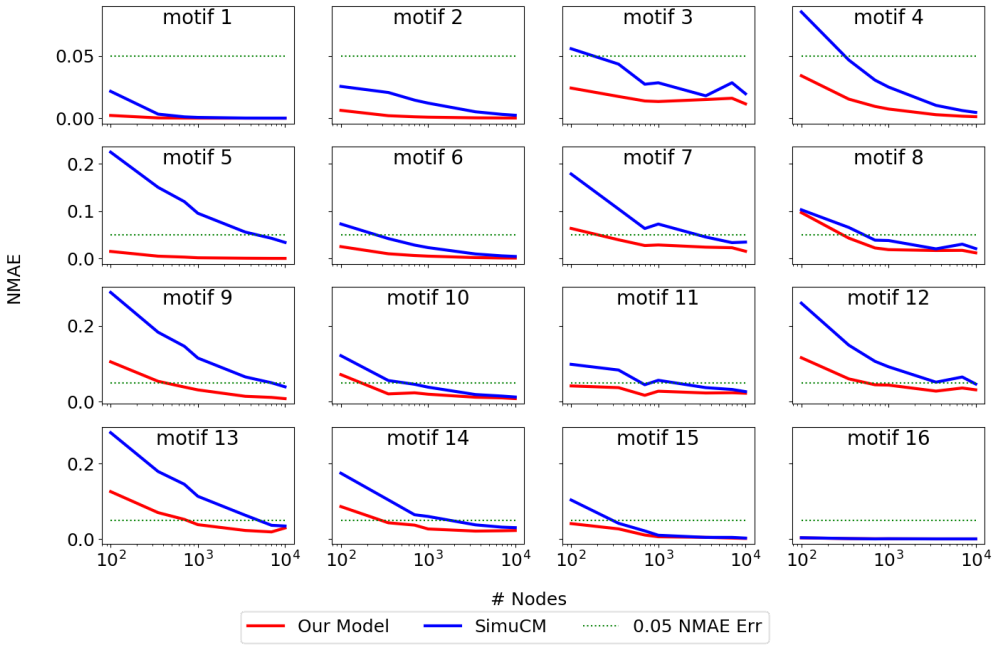


Figure 7: Network size and NMAE of our model and simuCM for BA graphs.

Table 3: Dataset Statistics

Datasets	Nodes	Edges	Avg Degree	# Degree pair
MathOverflow	11k - 21k	77k- 227k	4.17 -9.24	452 - 968
askUbuntu	74k - 153k	158k - 541k	1.98 - 12.86	589 - 1738
p2p-Gnutella	6k - 11k	20k - 88k	2.4 - 3.67	36 - 335
EmailEU	89 - 986	542 - 25k	9.24-25.28	26-211
EmailEnron	182	3010	16.53	60
socBitcoin	4k - 6k	24k - 36k	6.05 - 6.35	535 -554
WikiVote	7k	103k	14.67	457

6.3.3 Results

Figure 8 shows the KL-divergence from triad distribution of our method and simulation based method with configuration model (SimuCM) to baseline methods. The KL-divergences from our model are below 0.01, suggesting the triad distribution from our model are closer to baseline methods on all tested datasets.

6.3.4 Application in Network Clustering

One important application of motif census with a null model with BDS in real-world directed networks is to measure topological similarities and identify network types by clustering superfamily of networks [15] (a superfamily is defined as a type of network such as biological or email communication networks). A vector containing subgraph ratio profiles (SRP) for 16 triads are computed in each network and the similarity between networks is defined as the cosine similarity of the vectors.

The subgraph ratio profile (SRP) [15] for a motif i is defined as

$$SRP_i = \frac{\Delta_i}{\sqrt{\sum \Delta_i^2}}, \tag{22}$$

where Δ_i is a normalized term that measure the difference between the frequency of motif i in an empirical network (denoted as $N_{observed_i}$) and the average frequency in random networks in a null model (denoted as

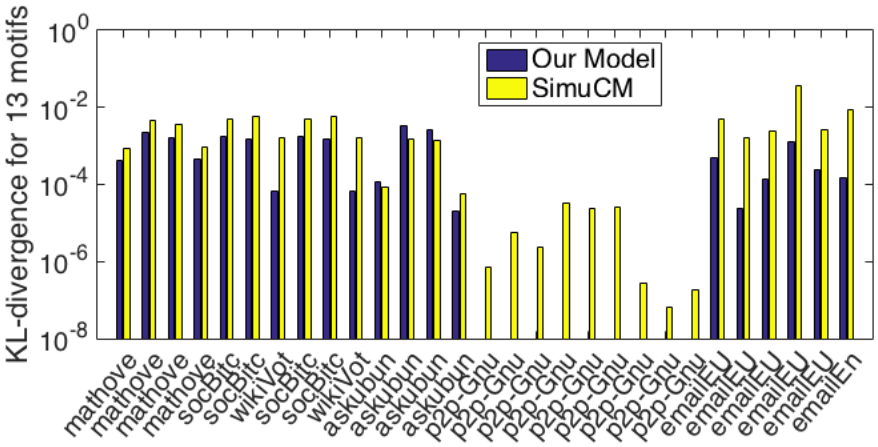


Figure 8: KL-divergence from distribution of 13 triads of our model and SimuCM to baseline method on real-world networks of different types and sizes.

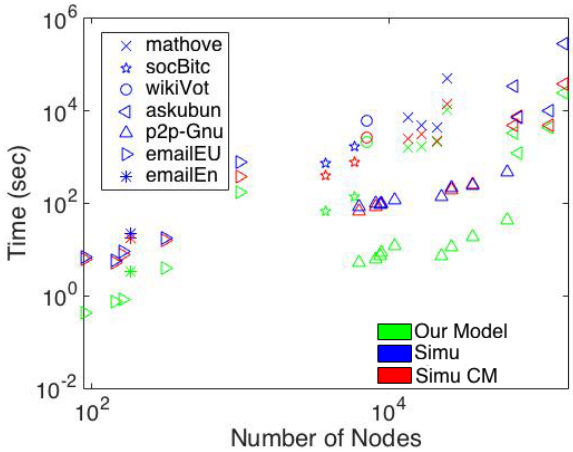
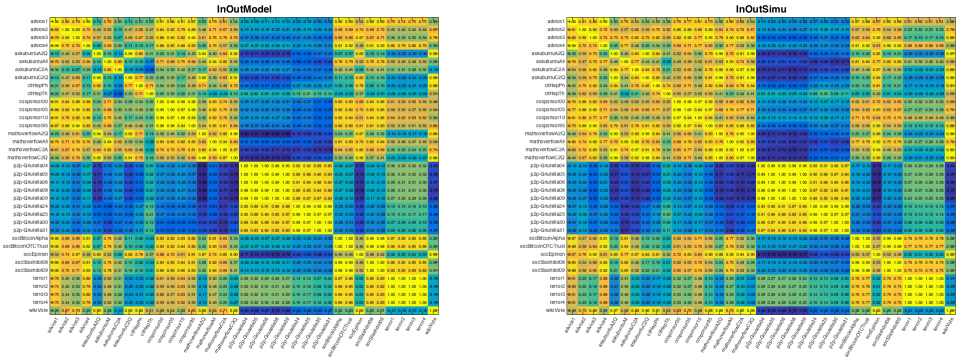


Figure 9: Runtime for real-world networks. Our method runs on average 10X time faster than simulation based methods



(a) Our Model (b) Simulation as Baseline

Figure 10: Visualization of correlation coefficient matrices of the triad SRPs for real-world networks. Light yellow represent 1.0, meaning two networks are similar while dark blue reprensent -1.0. The matrix form our model is close to the baseline and network clustering result from these two matrices are identical.

$< N_{random_i} >$):

$$\Delta_i = \frac{N_{observed_i} - < N_{random_i} >}{N_{observed_i} + < N_{random_i} > + \epsilon}, \tag{23}$$

where ϵ (usually set to 4) is an error term to make sure that Δ_i is not too large when the motif rarely appears in both the empirical and random graphs.

Clustering result of real-world networks with SRPs from our model is exactly the same as the simulation based model. Figure 10 shows the visualization of correlation coefficient matrices of SRPs for all real-world network.

7 Conclusion

In this paper, we propose a probability model to approximate the expected triad census in large directed random graphs with degree sequence. We design an fast algorithm for this model and gain speed much faster than simulation based methods. When the second moment of node degree is finite, the approximation error of triad frequency in our model is a vanishing $O(\frac{1}{N^2})$ fraction of the total number of the triads as network size N grows to infinity. Experiments with synthetic BA, ER and smallworld networks show that the time cost in our algorithm grows much slower than the baseline methods with the increase of the network size. The normalized mean absolute error (NMAE) of triad frequencies reduces within a $\pm 5\%$ error level before the network size reach 1000, suggesting our algorithm is suitable for large networks. Experiments with realwold networks show that our model can be applied to analyze networks of different types and obtain Compared to simulation based methods, our algorithm gains an average of 10X speed up on our realworld network datasets. KL-divergence from triad distributions in our model to those in baseline models are low. We apply our model to network clustering using SRP and obtain the same result as the based line method, suggesting our model obtain a good approximation and useful in real-wold application.

A Probability of Subgraph Structure given In/Out Degree

In a configuration model for a directed graph G with $|E|$ edges, u, v and w are three nodes in G, with joint in/out degree pairs $(I_u, O_u), (I_v, O_v)$ and (I_w, O_w). The probability of an edge from u to v can be approximated as $\frac{O_u I_v}{|E|}$. The probability that a subgraph of u, v, w has the i-th structure in Fig.1 can be

computed as:

$$P(\mathcal{T}_{u,v,w}=i)\approx\begin{cases}(1-\frac{O_uI_v}{|E|})(1-\frac{O_uI_w}{|E|})(1-\frac{O_vI_u}{|E|})(1-\frac{O_vI_w}{|E|})(1-\frac{O_wI_u}{|E|})(1-\frac{O_wI_v}{|E|}) & \text{if } i=1\\ \frac{O_uI_v}{|E|}(1-\frac{(O_u-1)I_w}{|E|})(1-\frac{O_vI_u}{|E|})(1-\frac{O_vI_w}{|E|})(1-\frac{O_wI_u}{|E|})(1-\frac{O_w(I_v-1)}{|E|}) & \text{if } i=2\\ \frac{O_uI_v}{|E|}\frac{O_vI_u}{|E|}(1-\frac{O_uI_w}{|E|})(1-\frac{(O_v-1)I_w}{|E|})(1-\frac{O_w(I_u-1)}{|E|})(1-\frac{O_w(I_v-1)}{|E|}) & \text{if } i=3\\ \frac{O_uI_v}{|E|}\frac{(O_u-1)I_w}{|E|}(1-\frac{O_vI_u}{|E|})(1-\frac{O_v(I_w-1)}{|E|})(1-\frac{O_wI_u}{|E|})(1-\frac{O_w(I_v-1)}{|E|}) & \text{if } i=4\\ \frac{O_vI_u}{|E|}\frac{O_w(I_u-1)}{|E|}(1-\frac{O_uI_v}{|E|})(1-\frac{O_uI_w}{|E|})(1-\frac{(O_v-1)I_w}{|E|})(1-\frac{(O_w-1)I_v}{|E|}) & \text{if } i=5\\ \frac{O_uI_w}{|E|}\frac{O_vI_u}{|E|}(1-\frac{(O_u-1)I_v}{|E|})(1-\frac{(O_v-1)(I_w-1)}{|E|})(1-\frac{O_w(I_u-1)}{|E|})(1-\frac{O_wI_v}{|E|}) & \text{if } i=6\\ \frac{O_uI_v}{|E|}\frac{O_vI_w}{|E|}\frac{O_w(I_v-1)}{|E|}(1-\frac{(O_u-1)(I_w-1)}{|E|})(1-\frac{(O_v-1)I_u}{|E|})(1-\frac{(O_w-1)I_u}{|E|}) & \text{if } i=7\\ \frac{O_vI_u}{|E|}\frac{(O_v-1)I_w}{|E|}\frac{O_wI_v}{|E|}(1-\frac{O_u(I_v-1)}{|E|})(1-\frac{O_u(I_w-1)}{|E|})(1-\frac{(O_w-1)(I_u-1)}{|E|}) & \text{if } i=8\\ \frac{O_vI_u}{|E|}\frac{(O_v-1)I_w}{|E|}\frac{O_w(I_u-1)}{|E|}(1-\frac{O_uI_v}{|E|})(1-\frac{O_u(I_w-1)}{|E|})(1-\frac{(O_w-1)I_v}{|E|}) & \text{if } i=9\\ \frac{O_uI_v}{|E|}\frac{O_vI_w}{|E|}\frac{O_wI_u}{|E|}(1-\frac{(O_u-1)I_w}{|E|})(1-\frac{(O_v-1)(I_u-1)}{|E|})(1-\frac{(O_w-1)(I_v-1)}{|E|}) & \text{if } i=10\\ \frac{O_uI_v}{|E|}\frac{O_vI_u}{|E|}\frac{(O_v-1)I_w}{|E|}\frac{O_w(I_v-1)}{|E|}(1-\frac{(O_u-1)(I_w-1)}{|E|})(1-\frac{(O_w-1)(I_u-1)}{|E|}) & \text{if } i=11\\ \frac{O_uI_v}{|E|}\frac{(O_u-1)I_w}{|E|}\frac{O_v(I_w-1)}{|E|}\frac{O_w(I_v-1)}{|E|}(1-\frac{(O_w-1)I_u}{|E|})(1-\frac{(O_v-1)I_u}{|E|}) & \text{if } i=12\\ \frac{O_vI_u}{|E|}\frac{(O_v-1)I_w}{|E|}\frac{O_w(I_u-1)}{|E|}\frac{(O_w-1)I_v}{|E|}(1-\frac{O_u(I_v-1)}{|E|})(1-\frac{O_u(I_w-1)}{|E|}) & \text{if } i=13\\ \frac{O_uI_w}{|E|}\frac{O_vI_u}{|E|}\frac{(O_v-1)(I_w-1)}{|E|}\frac{O_w(I_u-1)}{|E|}(1-\frac{(O_w-1)I_v}{|E|})(1-\frac{(O_u-1)(I_v-1)}{|E|}) & \text{if } i=14\\ \frac{O_uI_w}{|E|}\frac{O_vI_u}{|E|}\frac{(O_v-1)(I_w-1)}{|E|}\frac{O_w(I_u-1)}{|E|}\frac{(O_w-1)I_v}{|E|}(1-\frac{(O_u-1)(I_v-1)}{|E|}) & \text{if } i=15\\ \frac{O_uI_v}{|E|})\frac{(O_u-1)I_w}{|E|}\frac{O_vI_u}{|E|}\frac{(O_v-1)(I_w-1)}{|E|}\frac{O_w(I_u-1)}{|E|}\frac{(O_w-1)(I_v-1)}{|E|} & \text{if } i=16\end{cases}\tag{24}$$

B Number of Triads with $D(u,v,w)$ in Graph

To compute the triad census in Eq(8), we also need to compute the frequency of a subgraph structure with in/out degree tuple $D(u,v,w)$, denoted as $c(u,v,w)$. Let N_u, N_v and N_w denote the numbers of nodes of in/out degrees $(I_u,O_u),(I_v,O_v)$ and (I_w,O_w), respectively:

$$c(u,v,w)=\begin{cases}N_uN_vN_w & (I_u,O_u)\neq(I_v,O_v),\,(I_u,O_u)\neq(I_w,O_w),(I_v,O_v)\neq(I_w,O_w)\\ N_u(N_u-1)N_w & (I_u,O_u)=(I_v,O_v),(I_u,O_u)\neq(I_w,O_w)\\ N_uN_v(N_v-1) & (I_u,O_u)\neq(I_v,O_v),(I_v,O_v)=(I_w,O_w)\\ N_u(N_u-1)N_v & (I_u,O_u)=(I_w,O_w),(I_u,O_u)\neq(I_v,O_v)\\ N_u(N_u-1)(N_u-2) & (I_u,O_u)=(I_v,O_v)=(I_w,O_w)\end{cases}\tag{25}$$

C Generate Simple Graph by Rewiring Self-loops and Multi-Edges

For a directed graph generated from configuration model, we rewire the self-loop and multi-edges to reconstruct it to a simple graph. The algorithm is illustrated in Algorithm(2). Line 2 $\sim$ 6 detects selp-loops and multi-edges in the edge set E and add them to set S. For each edge $(u_i,v_i)\in S$, we sample an edge (a,b) uniformly from the rest of edges (Line 9) and make sure that rewiring $(u_i,v_i),(a,b)$ will not result in self-loops or multi-edges (Line 10), otherwise we re-sample a other edge (Line 11).

D Baseline Algorithms

Given a null model with a BDS, the ground truth of average frequency of a motif type is computation intractable. Current researches usually apply Algorithm 3 by computing the average frequency from a set of uniformly generated random graphs.

For large random graphs, We choose

Given a BDS, we choose two simulation based methods as baselines to compare to our methods: the first basline method (Algorithm 3) applies edge-rewiring technique to randomly generate simple graphs and compute the average frequencies for each triad; the second baseline method (Algorithm 4) impements a configuration model to generate random graphs, allowing self-loop and multi-edges.

Algorithm 2: Re-wiring Self-loops and Multi-Edges

Data: Edge Set $E = \{(u_1, v_1), (u_2, v_2), \dots\}$
Result: Edge Set E

$S \leftarrow \Phi$;
for $i = 1 : |E|$ **do**
  if $u_i = v_i$ ***or*** $u_i \in PAR(v_i)$ **then**
    $S \leftarrow S \cup \{(u_i, v_i)\}$;
  else
    $\text{PAR}(v_i) \leftarrow \text{PAR}(v_i) \cup \{u_i\}$
for $(u_i, v_i) \in S$ **do**
  while ***true*** **do**
    sample (a, b) uniformly from $E - \{(u_i, v_i)\}$;
    if $a = u_i$ ***or*** $a = v_i$ ***or*** $b = u_i$ ***or*** $b = v_i$ ***or*** $u_i \in PAR(b)$ ***or*** $b \in PAR(v_i)$ **then**
      continue
    else
      $w \leftarrow v_i$;
      $(u_i, v_i) \leftarrow (u_i, b)$;
      $(a, b) \leftarrow (a, w)$;
      break

Algorithm 3: Average Triad Census for Graph given Bi-Degree Sequence (BDS)

Data: BDS $B = [(I_1, O_1), \dots, (I_n, O_n)]$
Result: 16 avg triad frequences $[f_1, \dots, f_{16}]$

$S = \text{getUniqueBiDegreePair}(B)$;
for $(I_u, O_u) \in S$ **do**
  $p_u = (N_u, I_u, O_u) = \text{getNumOfNodeWBiDegree}(I_w, O_w)$
for $u = 1 : |S|$ **do**
  for $v = u : |S|$ **do**
    for $w = v : |S|$ **do**
      for $t = 1 : 16$ **do**
        $f_t \mathrel{+}= \text{getExpectedFreq}(t, p_u, p_v, p_w)$

E Traid Census Error Analysis

References

[1] Nesreen K Ahmed, Jennifer Neville, Ryan A Rossi, and Nick Duffield. Efficient graphlet counting for large networks. In *Data Mining (ICDM), 2015 IEEE International Conference on*, pages 1–10. IEEE, 2015.

[2] L. A. N. Amaral, A. Scala, M. Barthlmy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences*, 97(21):11149–11152, 2000.

[3] Albert-Laszlo Barabasi and Reka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[4] Peter Bloem and Steven de Rooij. Large-scale network motif learning with compression. *arXiv preprint arXiv:1701.02026*, 2017. MDL to count undirected motifs.

Algorithm 4: Average Triad Census for Graph given Bi-Degree Sequence (BDS)

Data: BDS $B = [(I_1, O_1), \ldots, (I_n, O_n)]$
Result: 16 avg triad frequences $[f_1, \ldots, f_{16}]$

```
S = getUniqueBiDegreePair(B) ;
for (I_u, O_u) ∈ S do
    p_u = (N_u, I_u, O_u) = getNumOfNodeWBiDegree(I_w, O_w)
for u = 1 : |S| do
    for v = u : |S| do
        for w = v : |S| do
            for t = 1 : 16 do
                f_t += getExpectedFreq(t, p_u, p_v, p_w)
```


[5] Ningyuan Chen and Mariana Olvera-Cravioto. Directed random graphs with given degree distributions. *Stochastic Systems*, 3(1):147–186, 2013.

[6] Ningyuan Chen and Mariana Olvera-Cravioto. Directed random graphs with given degree distributions. *Stochastic Systems*, 3(1):147–186, 2013.

[7] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[8] Pter L Erdos, Istvn Mikls, and Zoltn Toroczkai. A simple havel-hakimi type algorithm to realize graphical degree sequences of directed graphs. *Electronic Journal of Combinatorics*, 17(1):R66, 2010.

[9] Paul W Holland and Samuel Leinhardt. Local structure in social networks. *Sociological methodology*, 7:1–45, 1976.

[10] Hawoong Jeong, Blint Tombor, Rka Albert, Zoltan N Oltvai, and A-L Barabsi. The large-scale organization of metabolic networks. *Nature*, 407(6804):651, 2000.

[11] Janis Klaise and Samuel Johnson. The origin of motif families in food webs. *Scientific Reports*, 7(1):16197, 2017.

[12] Lauri Kovanen, Márton Karsai, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11):P11005, 2011.

[13] Mirjam Kretzschmar and Martina Morris. Measures of concurrency in networks and the spread of infectious disease. *Mathematical biosciences*, 133(2):165–195, 1996.

[14] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[15] Ron Milo, Shalev Itzkovitz, Nadav Kashtan, Reuven Levitt, Shai Shen-Orr, Inbal Ayzenshtat, Michal Sheffer, and Uri Alon. Superfamilies of evolved and designed networks. *Science*, 303(5663):1538–1542, 2004.

[16] Ron Milo, Nadav Kashtan, Shalev Itzkovitz, Mark EJ Newman, and Uri Alon. On the uniform generation of random graphs with prescribed degree sequences. *arXiv preprint cond-mat/0312028*, 2003.

[17] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[18] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45(2):167–256, 2003.

[19] Mark EJ Newman, Steven H Strogatz, and Duncan J Watts. Random graphs with arbitrary degree distributions and their applications. *Physical review E*, 64(2):026118, 2001.

[20] Ashwin Paranjape, Austin R. Benson, and Jure Leskovec. Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 601–610, 3018731, 2017. ACM.

[21] Tiziano Squartini and Diego Garlaschelli. Triadic motifs and dyadic self-organization in the world trade network. In *International Workshop on Self-Organizing Systems*, pages 24–35. Springer, 2012.

[22] Mika J Straka, Guido Caldarelli, and Fabio Saracco. Grand canonical validation of the bipartite international trade network. *Physical Review E*, 96(2):022306, 2017.

[23] Pinghui Wang, John CS Lui, Don Towsley, and Junzhou Zhao. Minfer: A method of inferring motif statistics from sampled edges. In *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*, pages 1050–1061. IEEE, 2016.

[24] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.