Computers & Industrial Engineering 119 (2018) 273–288

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

# Efficient simulation and analysis of mid-sized networks

Luis E. Castro, Xu Dong, Nazrul I. Shaikh*

*Department of Industrial Engineering, University of Miami, 283 McArthur Engineering Building, 1251 Memorial Drive, Coral Gables, FL 33146, USA*

ARTICLE INFO

*Keywords:*
Network simulation
Computational efficiency
Egocentric networks
Vectorization
Multi-core processing
Node sampling

ABSTRACT

There is growing interest in developing the abilities to simulate realistic social networks and analyze data generated from existing online social networks such as Facebook and Twitter. Amongst other things, researchers and practitioners need these abilities to study how opinions and information diffuse over networks and identify the influential agents in networks. However, the sizes of the social networks that need to be simulated and the amount of user generated data that needs to be analyzed is growing at a faster rate than the computational power of most of the modern day computers. This paper presents a memory efficient network representation and computational resource allocation algorithm that yields a scale-up of about 400; thus, given a constraint on the availability of computational resources, researchers can now use the proposed algorithm to simulate and analyze networks that are more than 100 times larger than what they could simulate otherwise. The proposed network representation is conducive to multi-core processing and random node sampling. Algorithms for computationally efficient execution of three random-node-sampling-based methods to estimate network metrics such as the network diameter and average path length are also presented in the paper. These algorithms yield a speed-up of about 40 even when the researcher requires a precision of more than 98%. The scale-up and speed-up numbers are based on a detailed performance analysis of the proposed algorithms that was conducted on synthetic networks of sizes ranging from 1000 to 1,000,000 nodes. The observed scale-up and speed-up performance of the proposed algorithms has been validated against the algorithms used in *igraph* and *statnet*-two popular network data analysis software package, and these results are also presented in this paper.

## 1. Introduction

There is growing interest amongst researchers and practitioners in developing the abilities to simulate realistic social networks and analyze data generated from real social networks such as Facebook and Twitter. However, the sizes of the social networks that need to be simulated and the amount of user-generated data that needs to be analyzed is growing at a faster rate than the computational power of modern day computers. There, therefore is a need to develop a system and a method for the simulation of large networks and the analysis of network data that is computational memory and time efficient. In this paper, we presents a memory efficient network representation and computational resource allocation algorithm to enable the simulation and analysis of large networks when the computational resources are limited.

The proposed network representation and computational resource allocation algorithm combines an egocentric list-based-representation of a network (Easley & Kleinberg, 2010; Goldenberg, Zheng, Fienberg, & Airoldi, 2010; Jackson, 2010; Newman, 2010) with efficient programming constructs such as vectorization and multi-core processing that function together and yield a scale-up of more than 400. The implication of such a scale-up is that computers that previously could barely handle the simulation and analysis of a network with 10,000–15,000 nodes can now support an analysis of a network with about 5,000,000 nodes. Such a scale-up reduces the need for researchers and practitioners to compromise on the scope, depth, and scale of their studies. The proposed algorithm is generalizable, has been validated, and now forms the backbone of the data representation, storage, and analysis methodology used within *fastnet* (https://cran.r-project.org/web/packages/fastnet/index.html), an R software package developed for the simulation and analysis of large networks.

The egocentric list-based representation of a network is conducive to multi-core processing and random-node sampling. Algorithms for the computationally efficient execution of three random-node-sampling-based algorithms that estimate various distance based network characteristics such as the network diameter and average path length are also presented. These algorithms yield a speed-up of about 40 even when the researchers requires a precision of more than 98%. Such a speed-up implies that it now takes approximately 15 min to estimate the average path length of a network with about 100,000 nodes on a 16 GB

* Corresponding author.
*E-mail address:* n.shaikh@miami.edu (N.I. Shaikh).

https://doi.org/10.1016/j.cie.2018.03.008
Received 24 January 2017; Received in revised form 9 February 2018; Accepted 4 March 2018
Available online 07 March 2018

RAM computer when the usual time requirement is about 10 h using algorithms available in the popular network data analysis software packages such as *igraph* (Csardi & Nepusz, 2006). A detailed performance analysis of the proposed network representation and the random-node-sampling-based algorithms using synthetic networks, and a comparison of the proposed algorithms against *igraph* and *statnet*, a popular network data analysis software package, using real networks is also presented.

The R language[1] has been used to develop and operationalize the proposed algorithms. R is popular with researchers and provides an easy and efficient way to integrate a wide variety of statistical packages (including many pre-built network/graph analysis libraries). All the proposed algorithms have been tested and incorporated into *fastnet* (Shaikh, Xu, & Castro, 2017), a network data analysis package in R.

In the next section, we present a brief review of literature and highlight some current trends with regards to network theoretical analysis in management science and epidemiology. In Section 3, we focus on the performance analysis of the proposed network representation and computational constructs that are integrated within the proposed algorithm to simulate and analyze networks. Section 4 contains the three sampling-based algorithms used to estimate key distance based network metrics. The performance analysis of these sampling-based algorithms in terms of speed and accuracy is presented in Section 5. A comparison of the proposed algorithms against algorithms used with *statnet* and *igraph* using data from three real networks is also presented in Section 5. The conclusions and suggestions for future research and development are presented in Section 6.

## 2. Network representation

Social network analysis (SNA) focus on large networks to (1) be able to simulate the behavior of a real system and analyze various what-if scenarios at faster rates, (2) study evolutionary behavior (Nicol, Liu, Liljenstam, & Yan, 2003; Yeom et al., 2014), and (3) to develop and test algorithms related to navigation (Chen, Wang, & Wang, 2010; Liu et al., 2014), clustering and partitioning (Handcock, Raftery, & Tantrum, 2007), community detection (Bickel & Chen, 2009; Karrer & Newman, 2011; Newman, 2006, 2013), or distributed computing (Fujimoto et al., 2003) on networks. Extant research in SNA (in the context of management and marketing science, and epidemiology) has been summarized in Tables A1–A4 in Appendix A.

The aforementioned research demonstrates that the topologies of large-scale networks can differ significantly from those of smaller systems that are used to serve as surrogates, and the true values of network-centric metrics, such as degree distribution and average path length, often are not preserved in subgraphs (e.g., Ebbes, Huang, & Rangaswamy, 2016; Lee, Kim, & Jeong, 2006; Newman & Watts, 1999). Research also shows that the dynamics of diffusion or the evolution of behavior over large-scale networks may differ significantly from the dynamics over small-size networks, which in turn could influence the inferences made from the simulation analysis (Bhatt, Fujimoto, Ogielski, & Perumalla, 1998; Yeom et al., 2014). Thus, there are important disincentives for simulating and analyzing small networks.

However, as the research summarized in Table A1 demonstrates, the simulation and analysis of large networks is also not straightforward. The availability of large computational resources (usually parallel processors) and the expertise to use it efficiently (using distributed computing and efficient manage memory[2]) are prerequisites.

[1] R provides an environment for statistical computing; see https://www.r-project.org/.

[2] The terms memory, primary memory, and random access memory (RAM) are used interchangeably.

### 2.1. Memory management on computers

It is imperative to manage the available memory on a computer to process large networks. Efficient memory management for network analysis can be achieved through: (1) improved information representation, (2) use of efficient operational (primarily programming) constructs, and (3) efficient memory handling during intermediate processes/computations.

#### 2.1.1. Information representation

A network is a collection of nodes and links. The problem of information representation in a network involves efficient storage and retrieval of the node and link information. In a network, nodes usually represent resources or economic agents. Of their various attributes, the state of its existence is key (e.g., is the node an adopter or a potential adopter?). This information can be stored as a vector or as a record/data frame. The memory requirements for representing the node information usually scale linearly with the number of nodes $N$. The links represent the connections (pathways for information flow) on networks. The number of possible links in a network scale quadratically with $N$; there can be up to $N \times (N-1)$ links in a network of size $N$ if the network is directed.

The existence of links, or the lack thereof, traditionally is represented by a $N \times N$ 0–1 "adjacency matrix" $A$, in which the element $a_{ij}$ provides information about the existence of a link between nodes $i$ and $j$. However, most large social networks are sparse, so allocating $N \times N$ memory units to store the link information can be inefficient. Significant efficiency gains in memory usage can be achieved by representing the adjacency matrix as a sparse matrix (Koenker & Ng, 2011). This sparse matrix representation replaces the use of an $N \times N$ adjacency matrix with three arrays that store information about non-zero matrix elements, column indices, and row indices (https://cran.r-project.org/web/packages/SparseM/index.html). Effective memory usage is significantly smaller than the number of memory units required to store information about the entire adjacency matrix. Butts (2008a) developed the *network* package in R for a sparse matrix–based representation of large and sparse networks and report a 99.8% reduction in memory requirements when an adjacency matrix of a network with 100,000 nodes and 100,000 links is represented as a sparse matrix.

An alternative to the adjacency matrix–based approach to storing network information is an egocentric representation of a network, wherein the focus is on individual nodes and their connections. Egocentric representation can be implemented in two ways: (1) using an array of size $N \times \overline{\overline{K}}$ (where $\overline{\overline{K}}$ is the maximum number of connections that any node can have in the network) or (2) by using a list with $N$ rows, with each row having a variable size (i.e., variable number of columns) dictated by the exact number of connections for a specific node. If the average number of connections is $\overline{K}$, the typical storage requirement for the list scales as $N \times \overline{K}$. These two approaches to egocentric representation have significantly different memory requirements that depend on $N$ and the ratio between $\overline{\overline{K}}$ and $\overline{K}$; the list is more efficient when the ratio between $\overline{\overline{K}}$ and $\overline{K}$ is larger than about 1.05 (an array of size $N \times \overline{\overline{K}}$ is usually more memory efficient than a list of size $N \times \overline{K}$).

#### 2.1.2. Operational constructs

Today's computers provide system-level features such as vector processors, multiple cores, and virtual memory to manage swapping and paging between primary and secondary memory to improve system performance. However, the onus is generally on the researcher to apply these performance enhancers. Two key operational constructs that employ these system-level features and are relevant to network analysis are vectorization and multi-core processing.

- ***Vector Processors and Vectorization:*** Vectorization is the process of rewriting a repeated statement (usually a loop) on a vector of size $N$