Distributed Generation of Mobile Call Graphs with DPLN Degree Distribution

Kyrylo Chykhradze Ivan Malyshev
Denis Turdakov Anton Korshunov

Ivannikov Institute for System Programming of the RAS

Ivannikov ISPRAS Open, Moscow, 2017

Outline

1. Motivation
2. Introduction
3. Background
4. Method
5. Evaluation
6. Conclusions

Motivation

Why do we need random graphs?

- One needs to test network analysis algorithms on huge graphs
- Modern networks reach several hundred million nodes
- Real data gathering requires huge time and resource costs; lack of datasets

Mobile Call Graphs (MCG):

- **nodes** — mobile cell users
- **edges** — interaction between users (calls, SMS, etc)
- **edge weight** — number of calls or its duration

Properties of MCG:

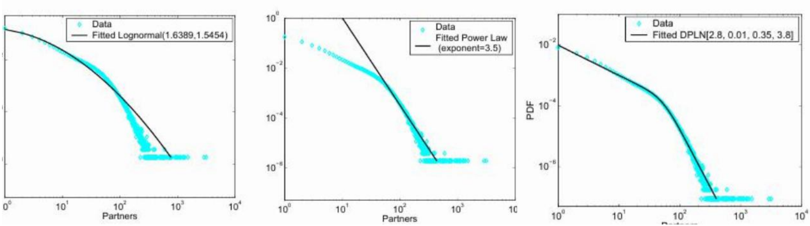
- **Degree distribution**
  - either Power Law distribution: $P(d) \sim d^{-\beta}$
  - or Double Pareto LogNormal distribution: $P(d) \sim DPLN(d, \alpha, \beta, \nu, \tau)$
- **Weights**
  - Good correlation between call duration and its amount
  - Distributed according to DPLN
- **Clusters and Clustering**
  - Heavy-tailed membership and community size distribution
  - High clustering coefficient

- Pareto: $f(x) = \begin{cases} \frac{\beta\, x_m^\beta}{x^{\beta+1}} & x \geq x_m \\ 0 & x < x_m \end{cases}$
- LogNormal:
  $X \sim LogN(\mu, \sigma^2) \Leftrightarrow \ln X \sim \mathrm{N}(\mu, \sigma^2)$
- **Double Pareto LogNormal**:

$$f(x, \alpha, \beta, \nu, \tau) = \frac{\alpha\beta}{\alpha+\beta}\left[e^{\alpha\nu+\alpha^2\tau^2/2}x^{-\alpha-1} \times \right.$$
$$\left. \times\, \mathrm{F}\left(\frac{\log x - \nu - \alpha\tau^2}{\tau}\right) + x^{\beta-1}e^{-\beta\tau+\beta^2\tau^2/2}\,\mathrm{F}^c\left(\frac{\log x - \nu + \beta\tau^2}{\tau}\right)\right],$$

where $F(x)$ - cdf of $\mathcal{N}(0,1)$, $F^c(x) = 1 - F(x)$

DPLN-distribution in pictures[1]:



[1] M. Seshadr et al. „Mobile call graphs: beyond power-law and lognormal distribution“

Problem Statement

Devise a random graph generator which allows to generate:

- networks fulfilling main properties of the mobile networks (DPLN degree distribution, community structure, etc);
- large graphs in a reasonable time.

Generators:

- GEDIS Studio Call Detail Record Generator[2]
- cdr-gen[3]
- Synthetic Call Detail Record generator using Spark[4]

[2] http://www.gedis-studio.com/online-call-detail-records-cdr-generator.html

[3] https://github.com/mayconbordin/cdr-gen

[4] https://github.com/RealImpactAnalytics/cdr-generator

Random MCG models:

- Migration model[5]
- Treasure-Hunt[6]
- Lognormal multiplicative process[7]

Real datasets:

- Wu et al. dataset (SMS)[8]
- VAST 2008 (cell interactions)[9]

---

[5] Lambiotte et al. „Geographical dispersal of mobile communication networks“

[6] Nanavati et al. „On the structural properties of massive telecom call graphs: findings and implications“

[7] Seshadri et al. „Mobile call graphs: beyond power-law and lognormal distributions“

[8] http://www.pnas.org/content/107/44/18803.abstract

[9] https://www.cs.umd.edu/hcil/VASTchallenge08

Main steps:

- Two heavy-tailed sequences generation (membership and community size distributions)
- Communities generation
- Edges generation

Proposed method

Phase 1: heavy-tailed sequences generation

Parameters:

- $\alpha, \beta, \gamma, \nu$
- x_{min}, x_{max}, y_{min}, y_{max}
- N

- Need to find C such that: $\int_{x_{min}}^{x_{max}} C \cdot f(x)dx = 1$. In discrete case: $C = \left(\sum_{x_{min}}^{x_{max}} f(x)\right)^{-1}$
- $\mathbb{E}(X) = \sum_{x} Cf(x) \cdot x$
- $N_1 \cdot \mathbb{E}(X) = N_2 \cdot \mathbb{E}(Y)$
- for each $x \in [x_{min}, x_{max}]$ add to sequence list $Cf(x) \cdot N$ elements with value x:
- $\underbrace{[10, 9, 9, \ldots, 1, 1, 1]}_{N}$

- d_i^1 — number of communities i^{th} user belongs to.
  d_j^2 — size of j^{th} community.
- from each sequence $[d_1, d_2, d_3, \dots]$ we create $[d_1, d_1 + d_2, d_1 + d_2 + d_3, \dots]$

- $D_k^1 = \sum_{i=1}^{k} d_i^1 \qquad D_k^2 = \sum_{j=1}^{k} d_j^2$
- $M = N_1 \cdot D_{N_1}^1$
- $x \sim u[1, D_{N_1}^1]$
  $y \sim u[1, D_{N_2}^2]$
- $p : x \in (d_p^1, d_{p+1}^1]$
  $q : y \in (d_q^2, d_{q+1}^2]$
- (p, q) — user p belongs to community q

Parameters:

- α, β, γ
- $\delta_{in}, \delta_{out}$

To choose node v according to $d + \delta$ ($v \sim d(v) + \delta$) in graph $G(t)$ means: $P(v = v_i) = \frac{d(v_i)+\delta}{t+\delta \cdot n(t)}$, $d(v_i)$ is a degree of v_i and $n(t)$ is a number of nodes at the time t.

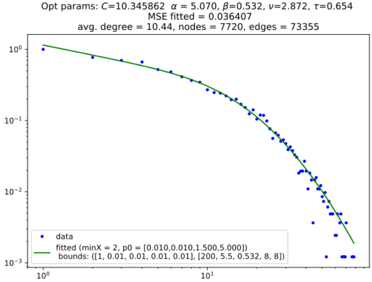
- G_0 is a graph without edges with a single node; $t_0 = 0$
- At the time stemp t graph $G(t+1)$ is created:
  1. $P = \alpha$: $w \sim d_{in} + \delta_{in}$
     new v; add $(v \to w)$
  2. $P = \beta$: $w \sim d_{in} + \delta_{in}$
     $v \sim d_{out} + \delta_{out}$
     add $(v \to w)$
  3. $P = \gamma$: $v \sim d_{out} + \delta_{out}$
     new w; add $(v \to w)$

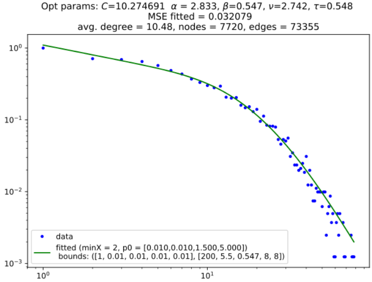
Weights of generated edges could be:

- generated randomly from Power Law distribution;
- equal to number of common membership of two nodes;
- unspecified (unweighted case).

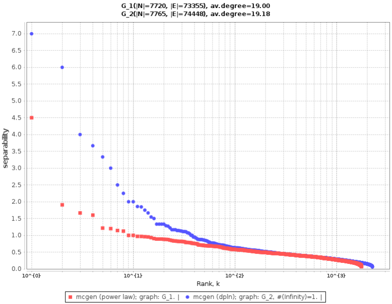
Accuracy Evaluation

Degree distribution

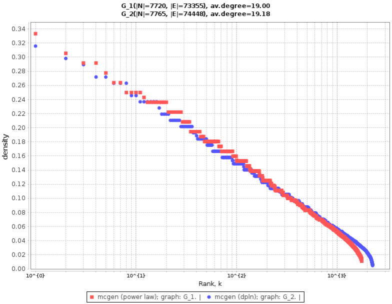




Separability



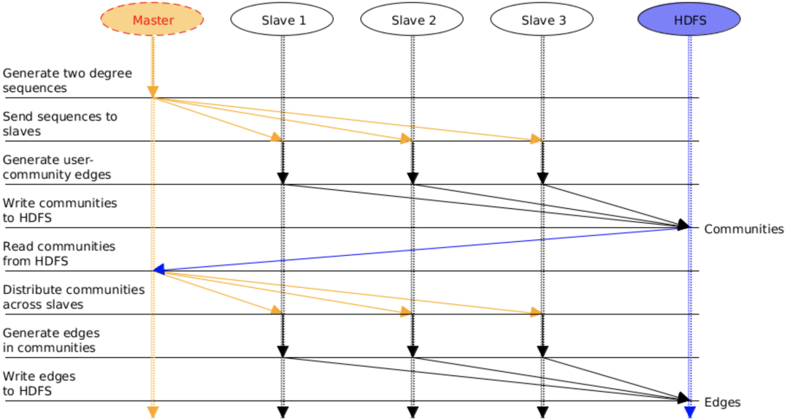
Density



$$g(S) = m_s / c_s \qquad f(S) = \frac{m_s}{|S|(|S|-1)}$$

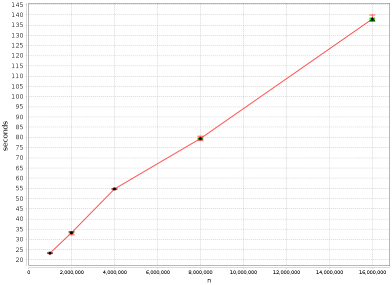
$$m_s = \sum_{i,j \in S} w_{ij} \qquad c_S = \sum_{i \in S, j \in \overline{S}} w_{ij}$$

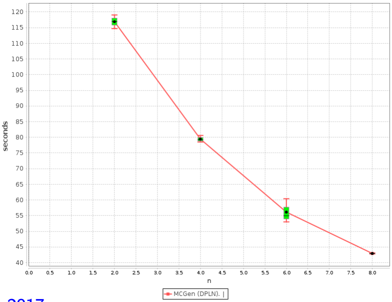
Performance Evaluation



Performance Evaluation

- For scalability by number of nodes (left) we used 4 `Spark.2xlarge` instances (64Gb, 8 CPUs) nodes.
- For scalability by number of slaves (right) we generated graph with $N = 8 \cdot 10^6$ on 4 `Spark.xlarge` instances (32Gb, 4 CPUS).





- Introduced new method for mobile call graph generation
- Method shows near-linear scalability both on the number of slaves and on the number of nodes
- Extended community statistics for weighted and directed case

Questions?
chykhradze@ispras.ru