Quantum Random Walk in Complex Networks

K. Avrachenkov, P. Jacquet, J. K. Sreedharan
k.avrachenkov@inria.fr, philippe.jacquet@alcatel-lucent.com, jithin.sreedharan@inria.fr

**Motivation from Community Detection**

- A classical problem in Graph theory
- It is more difficult when the graph is not known a priori
- An effective solution is Spectral Clustering:
  - For partitioning into k communities, it requires k upper extreme eigenvalues \( \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k \) and corresponding eigenvectors \( \mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k \)
  - Our aim is to find exact values or approximations of \( \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k \) and \( \mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k \) in a quicker way.

**Our Central Idea**

A variation of Spectral theorem says that

\[
\int_{-\infty}^{\infty} e^{iAt} e^{-i\theta t} dt = \sum_{j} \delta_{\lambda_j}(\theta) \mathbf{u}_j \mathbf{u}_j^T
\]

But to smooth the harmonic oscillations that will otherwise hide the Dirac peaks, and for ease of calculations we modify it to

\[
\int_{-\infty}^{\infty} e^{iAt} \mathbf{b}_0 e^{-i2\nu t} e^{-i\theta t} dt = \sum_{j} \sqrt{\frac{2\pi}{\nu}} \exp\left(-\frac{(\lambda_j - \theta)^2}{2\nu}\right) \langle \mathbf{u}_j \rangle \mathbf{b}_0 \mathbf{u}_j,
\]

where \( \mathbf{b}_0 \) is any initial vector.

**Computation with Quantum Random Walk**

Continuous time Quantum Random Walk (QRW) is defined as \( \psi_t = e^{-iAt}\psi_0 \),

where \( \psi_0 \) is a complex amplitude vector \( \{\psi_i(0), 1 \leq i \leq n\} \) with the probability of finding the QRW in node \( i \) at time \( t \) is \( |\psi_i(t)|^2 \)

The central equation (1) can be approximated in discrete time as

\[
\int_{-\infty}^{\infty} e^{iAt} \mathbf{b}_0 e^{-i2\nu t} e^{-i\theta t} dt \approx \mathbf{b}_0 + 2 \max_{\ell \leq 1} \sum_{\ell=1}^{\infty} e^{i\ell \theta} \langle \mathbf{u}_\ell \rangle \mathbf{b}_0 e^{-i\ell \theta} e^{-i2\nu t} (2)
\]

Using QRW for calculating extreme eigenvalues and eigenvectors:

1. Initialize QRW, \( \psi_0 \) with a random vector \( \mathbf{b}_0 \) in \([0, 1]^n\)
2. Sample the QRW at \( (\ell \nu) \) instants \( 1 \leq \ell \leq \ell_{\text{max}} \)
3. Calculate the approximation (2) at each node
4. Find peaks in (2) and thus obtain eigenvalues and eigenvector components
5. Do not measure the quantum system until the calculations are over

**Computation with Classical Computer**

We compute approximation (2) with the following centralized and distributed techniques. The main task is to compute \( e^{iAt} \mathbf{b}_0 \).

Graph example: character network in Les Misérables novel.

Nodes are the characters and edges are formed if two characters appear in the same chapter. Figures show the plot at the node Valjean.

**Distributed Diffusion Approach**

Calculate \( e^{iAt} \mathbf{b}_0 \) via Order-1, 2 and 4 approximations in distributed and asynchronous way.

Distributed: Each node knows only its neighbors

Asynchronous: Each node does not have to keep track the diffusion timings of its neighbors

Fluid diffusion approach: The idea is to compute the coefficients of the polynomial in \( z, \sum_{\ell=1}^{\ell_{\text{max}}} (I + iAt)^\ell \) using fluid diffusion with initial fluid at node \( i \) as \( \mathbf{b}_0(i) \) For order-2 and 4, polynomial will change accordingly.

**Centralized Approach**

Adjacency matrix is known. Approximating \( e^{iAt} \) by

Order-1: \((I + iAt)^\ell \)
Order-2: \((I + iAt + \frac{1}{2}(iAt)^2)^\ell \)
Order-4: \((I + iAt + \frac{1}{2}(iAt)^2 + \frac{6}{6}(iAt)^3 + \frac{1}{24}(iAt)^4)^\ell \)

**Distributed Monte Carlo Approach for Order-1**

\( R_k \triangleq (I + iAt)^k \mathbf{b}_0, 0 \leq k \leq \ell_{\text{max}} \)

\( R_{k+1} = (I + iAt)R_k, R_0 = \mathbf{b}_0 \)

\( R_k = R_k + \mu \mathbf{D} \mathbf{P} \mathbf{R}_k \), \( \mu \): direction diagonal matrix with entries as the degrees of the nodes

\( \mathbf{P} \): transition probability matrix of classical Random Walk on graph

The \( \ell \)th component of \( R_{k+1} \), \( R_{k+1}(i) = R_k(i) + i\mu D_{\ell}(\mathbf{P}_{\ell}(\xi_{\ell})) \), \( \xi_{\ell} \): a randomly picked neighbor of node \( i \).

**Implementation via**

a) Monte Carlo gossip
b) Parallel Random Walk

---

Note: Figures and tables are omitted for brevity.