Scalable Top-K Structural Diversity Search

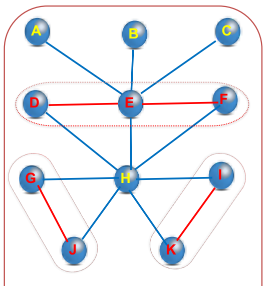
Lijun Chang[1], Chen Zhang[1], Xuemin Lin[1], Lu Qin[2]

[1]UNSW Sydney, Australia

[2]University of Technology Sydney, Australia

Problem Definition

- **Structural diversity** of a user in a social network is the number of connected components in its neighborhood, which measures the multiplicity of social contexts of a user since each connected component represents a distinct social context. (Ref. "*Structural Diversity in Social Contagion. PNAS'12. J. Ugander, L. Backstrom*")
- Given a threshold τ, the structural diversity $D_S(u)$ of u is the number of connected components, in the neighborhood-induced subgraph $G_{N(u)}$, whose sizes are at least τ. For example, $D_S(H) = 1$ if $\tau = 3$.
- **Problem Statement**: given a graph G and two integers k and τ, compute *k* vertices with the highest structural diversities according to the threshold τ.



A graph G and the neighborhood-induced subgraph $G_{N(H)}$ of H

State-of-the-art Approach [Huang et al. PVLDB'13]

- General Idea
  - an edge (v,w) is in $G_{N(u)}$ if and only if (u,v,w) forms a triangle in G.
  - $G_{N(u)}$ can be obtained by enumerating all triangles in G containing u, which computes $D_S(u)$.
- General Framework
  - **For** *vertices u in G in decreasing upper bound order*
    - ***If*** *the upper bound of u is no larger than the minimum of the current top-k results,* ***then break***
    - ***Else*** *compute $D_S(u)$ by enumerating triangles containing u, and update the current top-k result by u*
- **State of the art**, A*-B, dynamically tighten the upper bound of a vertex, and also use an A* search approach for testing whether $D_S(u) \geq \tau$ without actually computing the exact $D_S(u)$.
- **Drawbacks of** A*-B
  - A triangle (u,v,w) is enumerated three times, e.g., once in computing $D_S(u)$, $D_S(v)$, $D_S(w)$, respectively.
  - A hash table is constructed and probed for enumerating triangles, which incur non-negligible cost.
  - A hash table is also used for combining connected components in $G_{N(u)}$.

A Triangle Enumeration-based Approach

- General Idea
  - Adopt the state-of-the-art triangle enumeration algorithm, denoted TriE, for solving our problem, while enumerating each triangle at most once. (Ref. "*Triangle Listing Algorithms: Back From the Diversion. ALENEX'14. M. Ortmann, U. Brandes*")
- Challenges
  - To ensure each triangle is enumerated exactly once, TriE needs to process vertices in a specific order. However, for the efficiency consideration, we need to process vertices in decreasing order of their structural diversity upper bounds.
  - A triangle (u,v,w) is enumerated once but is needed for computing $D_S(u)$, $D_S(v)$, $D_S(w)$, and materializing triangles is space-consuming.
- Our Solution to Resolving the Challenges
  - We prove that by processing vertices in decreasing degree order, when processing a vertex u, we have generated all triangles containing u.
  - Rather than materializing triangles, we maintain the connected components of $G_{N(u)}$ for every vertex using the disjoint-set data structure, which takes linear space to the number of edges in G.
- **The Algorithm** Div-TriE
  - *Orient G to obtain a directed graph G^+, each edge pointing from the higher-degree vertex to the other vertex*
  - **For** *vertices u in G in decreasing degree (d(u)) order*
    - ***If*** *the upper bound ($d(u)/\tau$) of u is no larger than the minimum of the current top-k results,* ***then break***
    - *Enumerate triangles (u,v,w) such that $v,w \in N^+(u)$ by TriE, and update the connected components of $G_{N(u)}$, $G_{N(v)}$, $G_{N(w)}$.*
    - *Update the current top-k result by u.*

An Optimization Approach

- In Div-TriE, a hash table is still used to locate a connected component in a neighborhood-induced subgraph.
- We propose to associate the connected component containing v in $G_{N(u)}$ with edge (u,v) such that we eliminate the hash table. Denote the approach as Div-TriE*.
  - When enumerating triangle (u,v,w) with $v,w \in N^+(u)$ by TriE, we can directly locate the edges (u,v), (v,w), and (u,w), while the edges (v,u), (w,v), (w,u) are located by binding the two directions of each edge in an online preprocessing step.
- We propose techniques to bind the two directions of every undirected edge in $O(\alpha(G)\times m)$ time.

The time complexity of Div-TriE* is $O(\alpha(G)\times m)$.

Performance Studies

Comparing methods

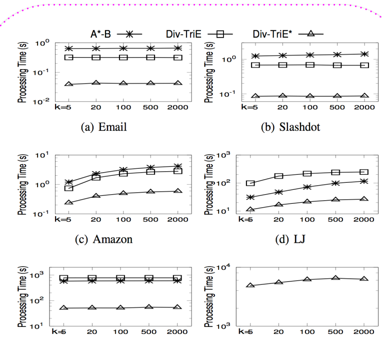
- A*-B: state-of-the-art approach
- Div-TriE: our triangle enumeration-based approach
- Div-TriE*: our optimization approach

Search Space: the number of vertices whose structural diversity are computed

Graph	Processing Time (seconds)			Search Space	
	A*-B	Div-TriE	Div-TriE*	Div-TriE*	A*-B
GrQc	0.016	0.017	***0.002***	974	***489***
CondMat	0.135	0.106	***0.013***	3,488	***1,465***
Email	0.648	0.315	***0.041***	4,270	***1,618***
Epinion	1.214	1.606	***0.114***	11,546	***4,887***
Slashdot	1.330	0.689	***0.085***	11,393	***5,559***
DBLP	1.096	1.030	***0.179***	10,726	***5,763***
Amazon	3.193	2.307	***0.493***	67,236	***30,198***
Google	4.352	5.083	***0.834***	40,536	***12,468***
Wiki	13.4	14.4	***2.643***	44,474	***14,590***
Skitter	25.8	29.6	***3.240***	47,866	***17,792***
LJ	71.9	217	***21.3***	121,084	***36,510***
KDD1	587	773	***51.0***	59,163	***5,331***
uk-2002	612	3215	***102***	146,899	**30,769**
Twitter	-	-	***6,160***	201,568	-

Compared with State-of-the-art Approach

k = 100, τ = 2



Vary k, τ = 2

Paper ID: 111