Information Flow and Search in Unstructured Keyword based Social Networks

Ankush Garg, **Prantik Bhattacharyya**, Charles U. Martel, S. (Shyhtsun) Felix Wu

UC DAVIS
UNIVERSITY OF CALIFORNIA

SMW '09, August 30, Vancouver, Canada

Introduction

- How to search information in online social networks (OSNs) ?

Searching in OSNs

- Expectations in terms of relevancy of result
  - results from direct friends
  - results from trustworthy friends

- Challenges in the absence of structure
  - high node degree
  - high clustering
  - low diameter

Search Problem

- Search a set of users with queried keyword as profile attributes
  - Output
    - Relevant results first
  - Constraints
    - Minimum resource usage

Outline

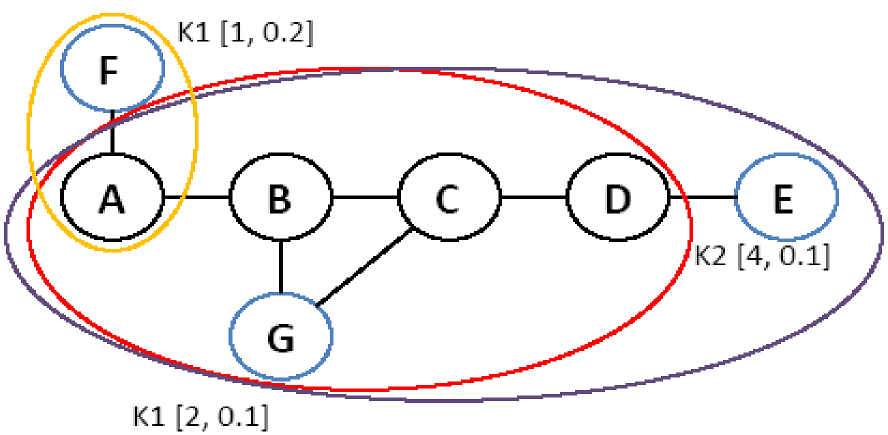
- Introduction
- OSN architecture
- Information Flow Model
- Search Algorithm
- Simulation Methodology
- Results
- Related Work
- Concluding Remarks

OSN architecture

- Undirected social network user topology
- No node has information about entire graph
- Identity of a node available to only friends
- Each edge has an associated trust value
- Each user *v* has profile attributes K_v^{PAtt}
- Profile attributes built by set of keywords
- Each keyword k has associated Policy(k)
- Policy(k) $= [\mathrm{D}, \mathrm{T}]$
- Each user *v* has friendly attributes K_v^{FAtt}

OSN architecture (more)

- D: maximum distance from where k is visible and *v* can be contacted from users
- T: min trust of each edge in social path required to let users contact *v*



- Each edge has trust > 0.5
- Each oval shows how keywords will flow and conversely, can be searched

Example: Keyword based social network

Information Flow Model

- Motivation
  - Information maintenance and update
  - Diffusion of non-inclusive keyword data
  - Integral to search algorithm development
- Users diffuse keyword information
  - Along with privacy consideration

Information Flow Model (more)

Algorithm 1: Processing Propagation Message

Input: v sends prop. message $\langle pid, K, hr, hc, T\rangle$ to u

if $((K \in FT_u)$ && $(pid \in FT_u))$ **then**
  if *new prop. data non-inclusive w.r.t stored prop. data corresponding to pid* **then**
    update $\{pid, hr, hc, T\}$ in FT_u;
    update search information;
    foreach *(friend z $(\neq v)$ of u)* **do**
      if $((hr > 1)$ && $(Trust_{uz} > T))$ **then**
        send $\langle pid, K, hr-1, hc+1, T\rangle$ to z;
        add/update z (for pid) in FT_u;
  else drop message;
else if $((K \in FT_u)$ && $(pid \notin FT_u))$ **then**
  foreach *(friend z $(\neq v)$ of u)* **do**
    if *new prop. data non-inclusive w.r.t all stored prop. data corresponding to z* **then**
      add $\{pid, hr, hc, T\}$ and z to FT_u;
      update search information;
      if $((hr > 1)$ && $(Trust_{uz} > T))$ **then**
        send $\langle pid, K, hr-1, hc+1, T\rangle$ to z;
    else drop message;
else
  add K and $\{pid, hr, hc, T\}$ to FT_u;
  update search information;
  foreach *(friend z $(\neq v)$ of u)* **do**
    if $((hr > 1)$&&$(Trust_{uz} > T))$ **then**
      send $\langle pid, K, hr-1, hc+1, T\rangle$ to z;
      add z (for pid) in FT_u;

- Hr: Hops remaining
- Hc: Hops covered
- PID; propagation ID to avoid cycles
- Identity of source suppressed to non-direct friends
- Non-inclusive data propagation
  - ΔHr > 0 || ΔT < 0 || ΔHc < 0

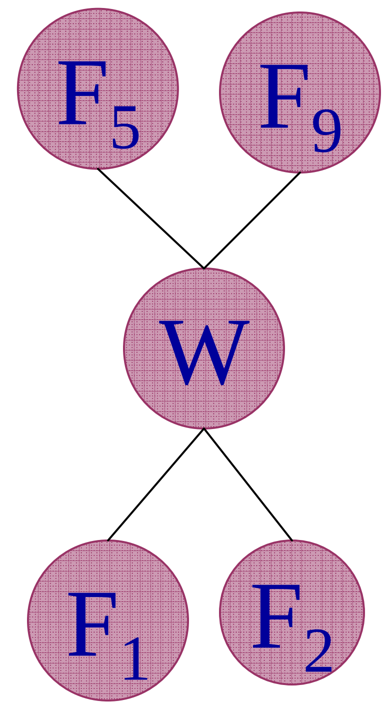
Information Flow Model (more)

- Keyword Forwarding Table, FT_w

Keyword	Propagation Data	Friends
K_1	$\{PID_1, Hr_1, Hc_1, T_1\}$	(F_1, F_2, F_3)

- Keyword Received Table, RT_w

Keyword	Max. Hops	(Friend, Min. Hops)
K_1	H_{max}^{K1}	$(F_9, Hmin^{K1}_{F9})$, $(F_5, Hmin^{K1}_{F5})$



Please see paper for details

Search Algorithm

- Fast lookup for boundary conditions
  - No results possible (no entry exists for query keyword)
  - Information provider are only the direct friends (Hmax == 1)
- Components:
  - Metric to define value of a direct friend
  - Threshold function for dynamic network pruning
  - Query message processing algorithm

Search Algorithm (more)

- For user w
  - Selecting Topologically Closer Nodes

$$\text{distance value}, DV(u, S_k) = \frac{\min_{k \in S_k} \mathrm{Hmax}^k - \max_{k \in S_k} H \min_u^k}{\min_{k \in S_k} H \max^k}$$

  - Selecting Trustworthy Nodes

$$\text{trust value}, TV(u, S_k) = \frac{T_{wu}}{T_{\max}^{S_k}}$$

  - Value for keyword set for a direct friend

$$V(u,S_k) = \rho \times DV(u,S_k) + (1-\rho) \times TV(u,S_k),\ 0 \le \rho \le 1$$

Search Algorithm (more)

- Threshold Function

$$\Theta(u, Q_k) = \max_{w \in N_u^{Q_k}} V(w, Q_k) - f(N_u^{Q_k}) \times (\max_{w \in N_u^{Q_k}} V(w, Q_k) - \min_{w \in N_u^{Q_k}} V(w, Q_k))$$

- Pruning function f with properties:
  - $f(1) = 1$ and $\lim_{N_u^{Q_k} \to \infty} f(N_u^{Q_k}) = 0$
- We use g(x) = x^{-p} for p ≥ 0 as f
  - p = 0, $f(N_u^{Q_k}) = \min_{w \in N_u^{Q_k}} V(w, Q_k))$
    - Breadth First Search (BFS)

Search Algorithm (more)

Algorithm 2: Processing Query Message

Input: v sends query message $\langle qid, Q_k, T_m, H_d, H_l \rangle$ to u

if $(qid \in QID_u)$ **then** drop message;

else

  add qid to QID_u;

  if $((Q_k \subseteq K_u^{PAtt})$ && $(\forall k \in Q_k,\ [H_d + 1, T_m]$ *satisfies* $Policy(k)_u))$ **then** send success message to v;   `/* u is a target */`

  if $(H_l > 0)$ **then**

    foreach *(friend* z $(\neq v)$ *of* u *such that* z *sent the keywords in* Q_k*)* **do**

      if $(Trust_{uz} < T_m)$ **then** $T_m \leftarrow Trust_{uz}$;

      if $(V(z, Q_k) \geq \Theta(u, Q_k))$ **then**

        send $\langle qid, Q_k, T_m, H_d + 1, H_l - 1 \rangle$ to z;

- Query message structure
  - <QID, Q_k, T_{min}, $Hops_{done}$, $Hops_{left}$>
- Starting Query Structure
  - <qid, Q_k, 1, 0, H_l>
- Query Serviced Table, QID_u:
  - <QID, direct friend who forwarded the query>

Search Algorithm (more)

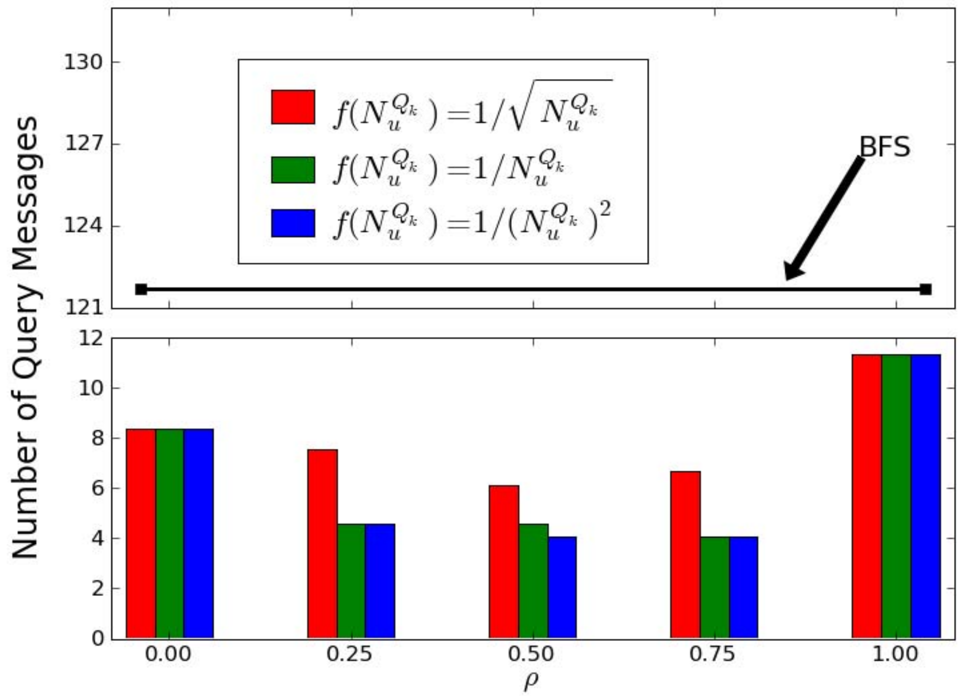
- Advantage of Dynamic Pruning
  - Reduces number of messages sent in network
  - Makes unstructured social network scalable
  - Selects targets of high value
    - topologically closer
    - and trustworthy
  - Returns a set of good results amongst all available obtainable results

Simulation Methodology

- Newman Watts Model
- Trust Distribution
  - Five Categories of Trust
    - From 'Blind Trust' (0.9) to 'Don't Know' (0.1)
- Information propagation policy
  - Restrictive policy
    - Depth set to 2 i.e. friends of friends
  - Liberal policy
    - Depth set randomly between 1 and graph diameter

Performance at ρ values

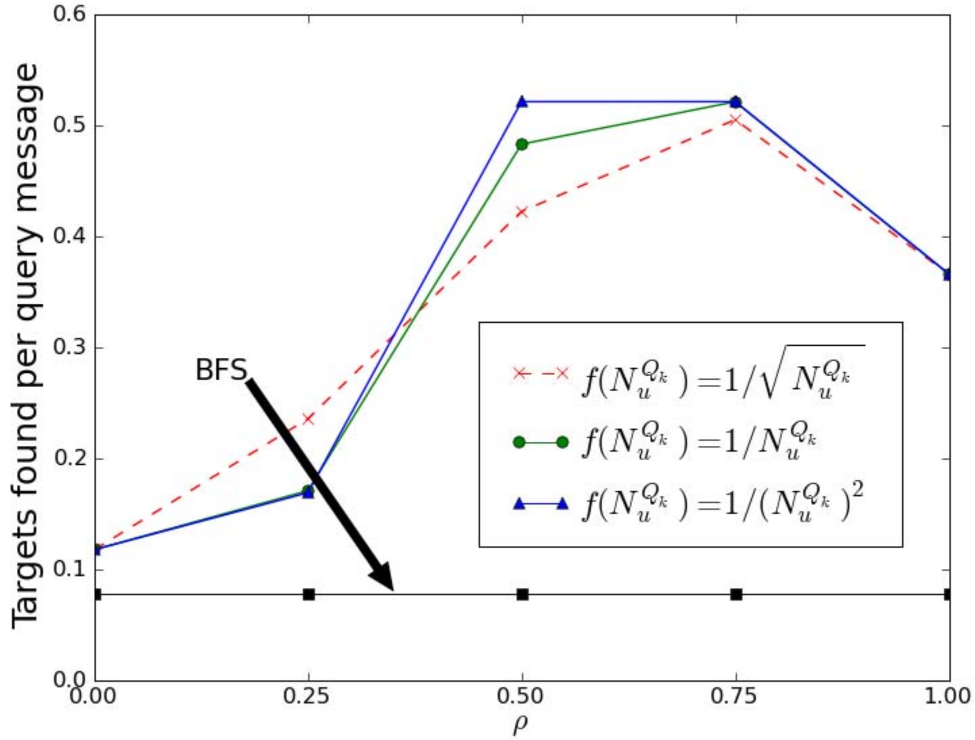
- Helps us understand system resource consumption
- Shows significant reduction in message generation when compared to BFS (12 compared to 121)



Analysis of network with restrictive propagation policy

Performance at ρ values (more)

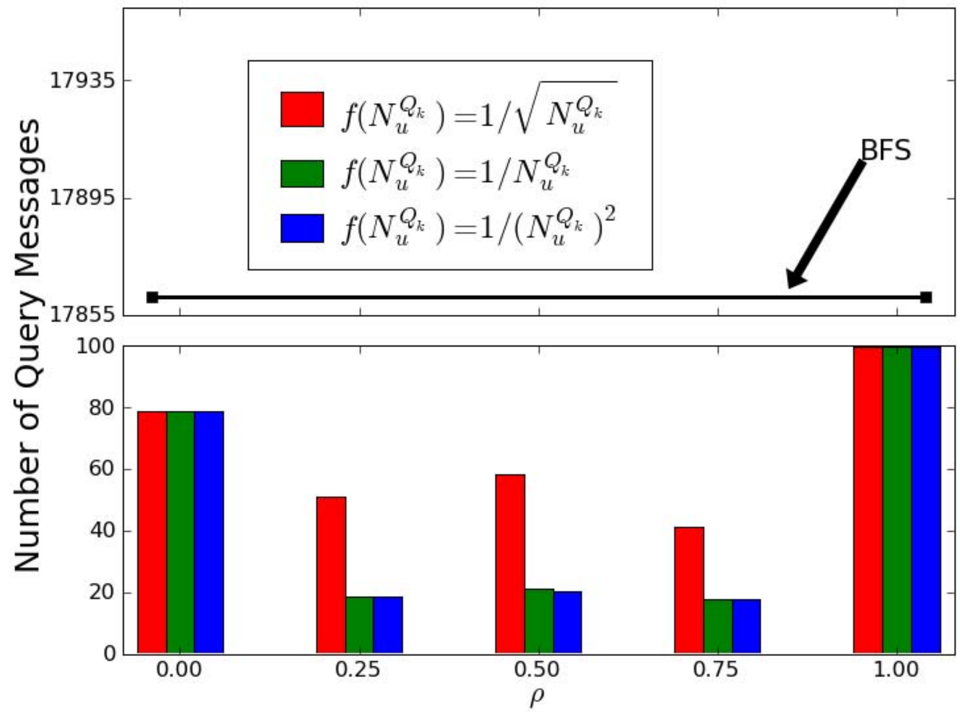
- Helps us understand how many queries are successful to find results
- Significant improvement at ρ = 0.5 when compared to BFS



Analysis of network with restrictive propagation policy

Performance at ρ values (more)

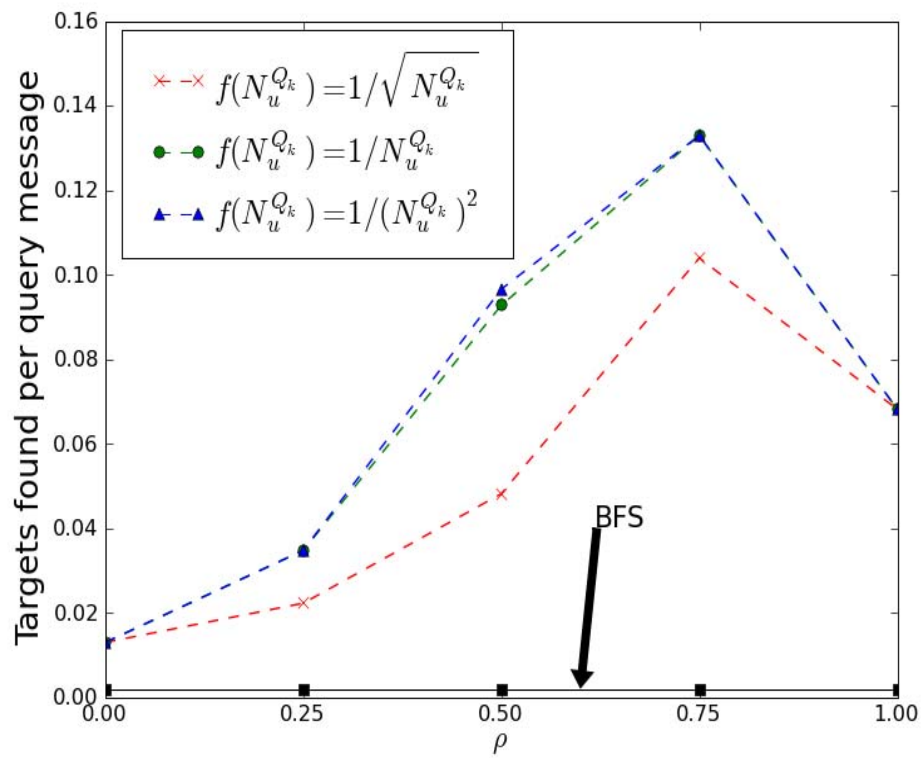
- Similar observation with higher levels of improvement
- Reduction in number of query messages generated from 17855 to only 100's..



Analysis of network with liberal propagation policy

Performance at ρ values (more)

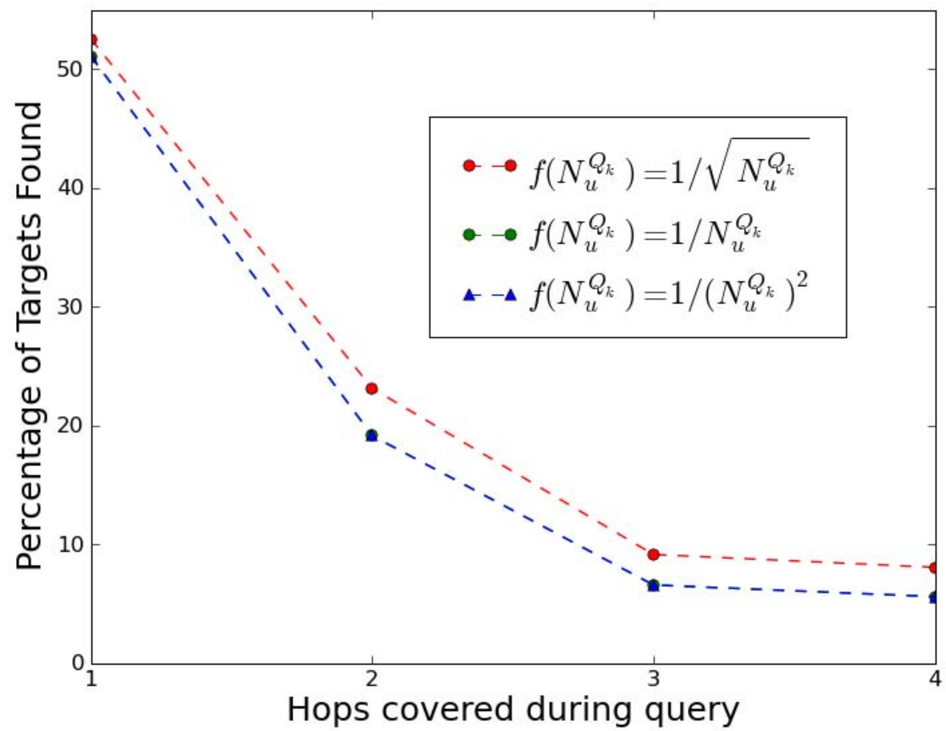
- Similar results with higher levels of performance
- Best results when hop and trust are considered together
- Results dip when either of the parameters are considered separately



Analysis of network with liberal propagation policy

Performance at hop values

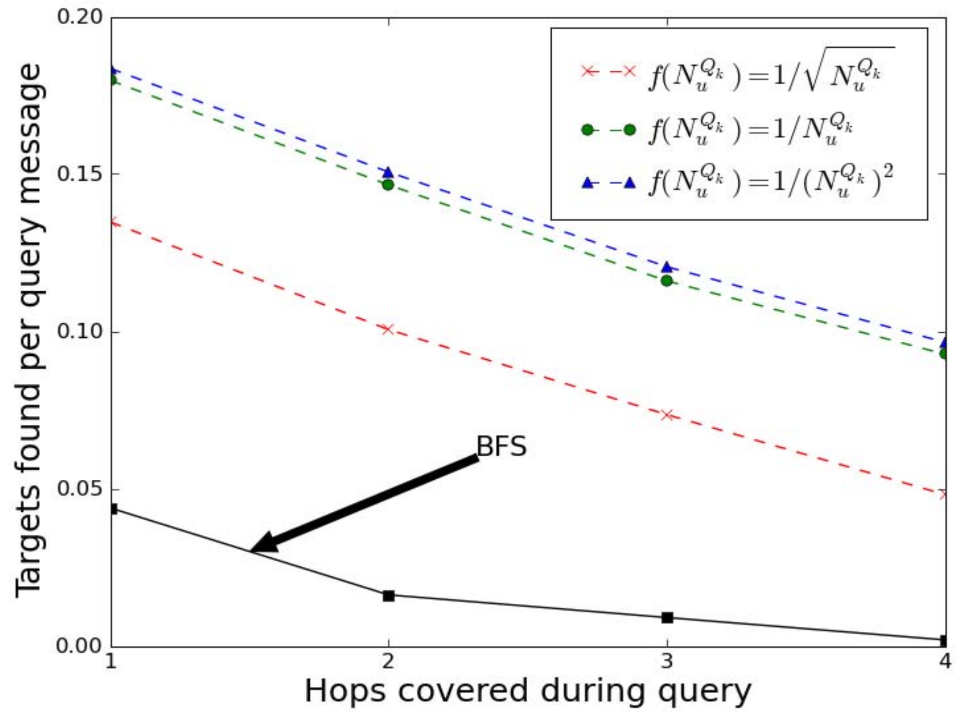
- Computed as a fraction of total number of results found by BFS
- Please see paper for more results



Analysis of network (liberal policy) for various hops and at ρ = 0.5

Performance at hop values (more)

- Higher denominator in pruning function associated with increased number of successful query messages
- Dynamic pruning using friend values is an effective way



Analysis of network with liberal propagation policy

Related Work

- Search in social networks
  - Algorithms using structural knowledge of the network through geographical distance, organizational hierarchy, interest of users..
- Search in decentralized and unstructured networks
  - BFS, random BFS, Intelligent BFS, directed BFS, iterative BFS, random walks, k random walks, other variation of random walks..

Concluding Remarks

- Developed a Search Algorithm
  - Using an Information Flow model
  - With focus on decentralization and privacy
  - Concentrates on finding set of good results
  - Dynamically prunes the network to search
  - Improvement in orders of magnitude
- Next step
  - Evaluation using larger graph topologies

Thanks!