Detecting Conversing Groups of Chatters: A Model, Algorithm and Tests

S. A. Çamtepe, M. Goldberg, M. Magdon-Ismail, M. S. Krishnamoorthy
{camtes, goldberg, magdon, moorthy}@cs.rpi.edu
Motivation and Problem

- Internet chatrooms are open for exploitation by malicious users
  - Chatrooms are open forums which offer anonymity.
  - The real identity of participants are decoupled from their chatroom nicknames.
  - Multiple threads of communication can co-exist concurrently.

- Our goal is
  - to provide automated tools to study chatrooms
  - to discover *who is chatting with whom?*
    - Human monitoring is possible but not scalable.
Motivation and Problem (cont.)

- Not a trivial task even for a well trained human eye

[20:19:40] <id1> what u been up to or down to?
[20:19:42] <id2> Hi!! Anyone from Boston around?
[20:20:00] <id3> amenazas d eataque
[20:20:04] <id3> sorry
[20:20:29] <id4> laying around sick all weekend...
[20:20:30] <id1> can anyone in here speak english??
[20:20:37] <id4> what about you ?
[20:20:40] <id1> whats wrong?
[20:20:46] <id5> not me
[20:20:48] <id4> sinus infection
[20:20:57] <id1> i had that last week
[20:20:58] <id6> Hmm, those seem rather infectious down there
[20:21:07] <id1> its this stupid weather
[20:21:32] <id4> yeah...darn weather
[20:21:43] <id1> i wont get good and over them until we get a good rain or a really hard freeze
Outline

- Related work
- Contributions
- The Model
- Algorithms
  - Cluster
  - Connect
  - Color & Merge
- Results
- Conclusion
IRC - Internet Relay Chat

- RFC 2810 … 2813
- Interactive and public forum of communication for participants with diverse objectives.
- IRC is a multi-user, multi-channel and multi-server chat system which runs on a Network.
- It is a protocol for text based conferencing:
  - Provides people all over the world to talk to one another in real time.
  - Conversation or chat takes place either in private or on a public channel.
Related Work

- **Multi-users in open forums**

- **An automated surveillance system**

- **PieSpy**
  - P. Mutton and J. Golbeck, “Visualization of Semantic Metadata and Ontologies”, IV03.

- **Chat Circle**

- **Social Network Analysis (SNA)**
Contributions

- A model which does not use semantic information,
  - chatters are nodes in a graph,
  - collection of chatters is a hyperedge,

- Two efficient algorithms
  - Uses statistical information on the posts to create candidate hyperedges,
  - “Cleans” the hyperedges using:
    - Transitivity,
    - Graph coloring,

- Algorithms are rigorously tested using simulation on the model.
The Model - Assumptions

- We model a single chatroom which corresponds to a topic.

- Members form small groups and talk on one or more subtopics:
  - Subtopics are created at the beginning and never halts.

- A user participates in one subtopic only. A user:
  - arrives,
  - selects a subtopic to talk on,
  - stays in the same subtopic during his/her lifetime.

- At any time, only one user is selected to post in a subtopic.
- Message interarrival times are random according to a given distribution.
The Model – Assumptions (cont.)

User’s arrival and departure times are selected uniformly at random. To make a user to post enough messages for analysis:

- Simulation time is divided into “n” regions
- Arrival times are selected uniformly at random from the first region,
- Departure times are selected uniformly at random from the last region.

At any time, messages coming from all subtopics are uniformly at random shuffled and output.
The Model - Parameters

- Simulation time and number of regions
- Number of users
- Number of subtopics
- Probability distribution and parameters (mean, variance, ...) for:
  - User to subtopic assignment
  - Message interarrival time
- Step size $K$ (will be defined in the next slide)
The Model - Algorithm

- Single event queue
  - Message post events (post, user, subtopic, time)
  - User join events (join, user, subtopic, time)
  - User leave events (leave, user, subtopic, time)

- K-step posting probability for each subtopic
  - A list of size $K$ named as “Probability History List”
  - A user who post recently is pushed to front
  - A user at the front has smallest probability of post next
  - A user at the end and users not in the list have the highest probability of post next
The Model - Algorithm (cont.)

- Load parameters
- For each user
  - select an arrival time, generate an arrival event for the user
  - select a departure time, generate a departure event for the user
  - select a subtopic, generate join event
- For each timestep
  - For each events of current time
    - If post event
      - insert the message to message buffer
      - create new post event
        - select next user to send according to K-step probability
        - select time for next post (message interarrival time)
        - update K-step probability (probability history list)
    - If join event
      - add user to subtopic
      - If first user in the subtopic,
        - create a post event
        - update K-step probability (probability history list)
    - If leave event
      - remove user from subtopic
  - Shuffle the message buffer
  - Log the messages to a file
The Model - Output

- Sample chat log
  - TIME 6 USER 20
  - TIME 7 USER 15
  - TIME 9 USER 61
  - TIME 12 USER 41
  - TIME 12 USER 24
  ......

- User to subtopic assignments

<table>
<thead>
<tr>
<th>Subtopic</th>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>20,41</td>
</tr>
<tr>
<td>3</td>
<td>61</td>
</tr>
<tr>
<td>4</td>
<td>24</td>
</tr>
</tbody>
</table>
Algorithms

- Initial processing of message logs
  - Consider every consecutive messages
  - Generate list of node pairs and the corresponding interarrival times

Sample Log

<table>
<thead>
<tr>
<th>TIME</th>
<th>USER</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>61</td>
</tr>
<tr>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>12</td>
<td>24</td>
</tr>
</tbody>
</table>

Node-pair, Interarrival list

- users (20,15) int. time 1
- users (15,61) int. time 2
- users (61,41) int. time 3
- users (41,24) int. time 0
Simple Clustering (K-Means)

- K-means clustering algorithm is applied on Interarrival list
- Generates two clusters
  - Red: pairs which has small interarrival times are put into this cluster
  - Blue: pairs which has large interarrival times are put into this cluster
Simple Clustering (K-Means)

- K-means clustering on Interarrival list
- Generates two clusters:
  - Red: pairs which has small interarrival times
  - Blue: pairs which has large interarrival times
- Declares:
  - Red pairs as not engaged in conversation
  - Blue pairs as engaged in conversation
- Idea: interarrival time between messages of two users, who exchanges messages over a subtopic, can not be smaller than a threshold:
  - It takes time for user to read, interpret, prepare answer
  - Network and servers introduce additional latency
Issues:

- Incomplete, it does not identify members of sub topics (conversing groups)
- May include contradictory information
  - For group of three users a, b, c
  - (a, b) and (a, c) are blue, (b, c) is red
  - Are (a, b, c) in the same subgroup???

Algorithms *Connect* and *Color_and_Merge*

- Reconcile possible contradictions
- Produce complete output
Algorithms – Connect

- Takes blue and red clusters
- Trusts blue cluster
- Considers blue cluster as the edge set of a graph $B$
  - Finds connected components in $B$
    - breath-first search on $B$
  - Consider previous example
    - For group of three users a,b,c
    - (a,b) and (a,c) are blue, (b,c) is red
    - Connect concludes that (a,b,c) are in the same subgroup.
Algorithms – Color

- Takes blue and red clusters
- Trusts red cluster more than blue cluster
- Considers red cluster as the edge set of a graph \( R \)
  - Applies vertex coloring
    - Uses heuristic Greedy to find an approximate solution
- Generates color classes
Algorithms – Merge

- Takes color classes generated by color
- For each pair of color classes $C_1$ and $C_2$
  - $e_b = \text{number of user pairs } (x, y) \text{ where}$
    - $(x, y) \text{ in blue cluster}$
    - $(x \text{ in } C_1 \text{ and } y \text{ in } C_2) \text{ or } (y \text{ in } C_1 \text{ and } x \text{ in } C_2)$
  - If $(e_b/|C_1|.|C_2| \geq \text{threshold}) \text{ merge } C_1 \text{ and } C_2$
- For our model, we found that threshold 0.7 gives good results
- Announce final color classes as subtopics.
Tests

- Parameters of the model are tuned according to observations and statistical analysis over real chatroom logs.
- A user pair which is announced correctly as being in the same subtopic is accepted as a correct result.
- Success rate = \( \frac{\# \text{ correct results}}{\text{all}} \)
- Following slide lists results for:
  - 5 topics, 50 users
  - 5 topics, 75 users
  - 10 topics, 50 users
  - 10 topics, 75 users
Results

- For sufficiently long log size, all algorithms converge to 100% success

- Critical factor is number messages per user.
  - As the number of users increases, larger logs are required

- Color_and_Merge algorithm provides the best result.
  - Converges to 100% success very quickly

- Connect is the most sensitive algorithm to log size

- As the log size decreases, connect fails faster
  - Why? A single false edge may connect two components yielding too much false results
Conclusion

- We presented a model for which we showed that it is possible to accurately determine the conversation.
- Ideas can be generalized to more elaborate models.
- Future work:
  - Enhance the model
    - Users may belong to multiple conversations
    - Users may switch between conversations
  - Apply algorithms to real chatroom logs