StackOverflow Community: The Top 5 Tags

Introduction
StackOverflow is a technical question-answer site that allows users to post questions to the community. Other members of the community make posts to answer these questions. Questions may be given descriptive tags (such as “c#”, “java”, “android”, etc.) to help improve search efficiency. Also, users are awarded “reputation” for being active on the site (either by asking/answering questions or making comments/edits, etc.). For this assignment, I focused primarily on question/answering patterns for the 5 tags with the most posting activity.

Dataset
The data was collected from the StackExchange Data Explorer (a public site that allows anyone to run arbitrary queries on the StackOverflow site). In particular, since only 50,000 rows can be retrieved at one time, only data regarding the top 5 tags were taken. The top 5 tags are: c#, java, php, javascript, and android. Also, since I was only interested in “strong interactions” (and did not want to compromise NodeXL’s performance too much) I only retrieved data for users who had interacted with another user at least 3 times. (E.g. User A posted an answer to at least 3 of User B’s questions). In this dataset I had 2201 vertices and 1869 edges.

Preprocessing
Initially, the data contained information regarding which tags each user was involved in. As expected, many users had contributed to multiple tags (e.g. asked a question tagged with multiple tags or answered questions with different tags, or both). However, NodeXL does not seem to support overlap between groups. So, to simplify grouping for NodeXL, I wrote a script to group Users by the tag the contributed to the most. (E.g. if User A contributed twice to “php” and three times to “android”, their dominant tag activity was in “android”)


Figures and Headlines

*Figure 1 - Initial graph*

**User's dominant tag activity:**

- **c#**
- **java**
- **php**
- **javascript**
- **android**

**Encodings:**
- Vertex = A user
- Edges = Indicate interaction between users (e.g. one user posted an answer to the other’s question)
- Color: The tag each user had the most activity in (see legend in Figure 1)
- Vertex Size: The user’s reputation (larger = higher reputation)
- Edge Width: The number of interactions between the two users (larger edges = more interactions, e.g. greater number of times one user posted answers to another user’s questions)

**Notes:**
- I used the above encodings and a Force-Directed layout (Fruchterman-Reingold) hoping to see vertices grouped closely together. However, the initial positions of vertices seemed to be unorganized so vertices with the same dominant tag were not clustered together after applying the layout. Hence, any relationships between groups were very unclear. So, I did not gain any useful insights from this visualization. However, the density of edges around certain nodes (E.g. green android user in the bottom left and dark blue c# user in the top right) does suggest that there are some users who interact more with other users.
Figure 2 - Grouping by Dominant Tag

Encodings:
- Vertex = A user
- Edge = Indicate interaction between users (e.g. one user posted an answer to the other’s question)
- Color: The tag each user had the most activity in
- Vertex Size: The user’s reputation (larger = more reputation)
- Edge Width: The number of interactions between two users (larger edges = more interactions)
- Shape: The 3 users with the most outgoing-edges (i.e. helped the most people) are diamonds
- Position: Vertices have been grouped by dominant tag

Notes:
- I selected the option to layout each group in its own box and applied a Force-directed layout within each box
- While the groups are now clearer, it still isn’t clear what the group relationships are.
- Also, no filtering was applied so the shapes were not very distinguishable
- As before, there are 2 users (diamonds in c# and android) who seem very involved in the community
Headline 1: Not all groups help each other equally

Figure 3 – Collapsing groups

Encodings:
- Vertex: group of users who were most active in each respective tag group
- Directed Edge: combined answering interactions between groups (points from answerer to asker)
- Color/Label: Dominant tag group
- Edge/Arrow width: Number of interactions between groups

Notes:
- After collapsing into groups, the overall interactions between tag communities are clearly not equal:
  - Tag groups related to traditional desktop application development (c# and java) help each other fairly equally. Similarly, tag groups related to web development (javascript and php) help each other fairly equally. This may be because the environments are similar so there is much common knowledge between the groups.
  - Interestingly, even though Java and Javascript are very different languages, java community members are able to help javascript community members quite a lot. This may suggest that Java members are more knowledgeable in general and can help users in other groups.
  - Lastly, it is of note that c# receives the most help from other communities, suggesting that it might be a common language on StackOverflow.
Figure 4 - Grouped Force-directed layout

Encodings:
- Vertex = A user
- Edge = Interaction between users
- Color: The tag each user had the most activity in
- Vertex Size: The user’s reputation (larger = more reputation)
- Edge Width: The number of interactions between two users (larger edges = more interactions)
- Shape: The 3 users with the most outgoing-edges (i.e. helped the most people) are diamonds
- Labels: The 10 users with the most outgoing-edges are labeled
- Position: Vertices have been grouped closer to vertices they interact more with

Notes:
- Figure 4 was created by starting with the vertices in the group-in-a-box layout and applying a force-directed layout on the whole graphical pane. The result of this was that vertices that were more involved with other vertices in the same group, tended to stay clustered together
- Something of note was that the labeled vertices tended to be clustered with other vertices in the same group (as opposed to vertices in other groups). This suggests that community experts tend to be more involved within just their tag group. Also, these top community contributors did not seem to interact much with one another.
Headline 2: Most leaders do not help each other, but are connected by people whose questions are answered by many people

*Figure 5 - Top contributor Connections*

- **Encodings:**
  - Vertex = A user
  - Directed Edge = One user posting an answer to another user's question
  - Color: The tag each user had the most activity in
  - Vertex Size: The user's in-degree (larger = more people answer their questions)
  - Edge Width: The number of interactions between two users (larger edges = more interactions)
  - Shape: The 3 users with the most outgoing-edges (i.e. helped the most people) are diamonds
  - Labels/Boxed: The 10 users with the most outgoing-edges are labeled and boxed in purple
  - Position: Vertices have been grouped closer to vertices they interact more with

- **Notes:**
  - Figure 5 was created from Figure 4 by filtering out nodes with low betweenness centrality
  - It is clear that most leaders have no direct links with one another. There is one exception when Sarfraz posted some javascript questions that were answered by Nick Craver (bottom right)
  - Also of note is that these leaders tend to be connected by users who receive answers from many people (maybe by asking more complicated questions or just more questions in general). This is not too surprising since if a user receives many answers, it is more likely that a top contributor gave an answer.
Headline 3: Users who answer many questions do not have many people answering their questions

*Figure 6 – Number of people I have helped vs. number of people who have helped me*

Encodings:
- Vertex = A user, Color: The tag each user had the most activity in
- Vertex Size: The user’s reputation
- Shape: The 3 users with the most outgoing-edges (i.e. helped the most people) are diamonds
- Position: the vertex’s x-axis position corresponds to that user’s in-degree (number of people who post answers to their questions). The vertex’s y-axis position corresponds to that user’s out-degree (number of people whose question that user has posted an answer to)

Nodes:
- This figure was created by using the above encodings, removing all filters, and hiding edges
- It is clear that the users who help the most people do not receive much help (relative to others). This suggests that these users are domain experts who do not ask many questions. Alternatively, they may ask very complex questions that other users are not able to answer.
- Similarly, users who ask the most questions do not provide many answers. These users might be novices who need more help and do not have the expertise to answer questions.
NodeXL Critique
Overall, NodeXL made it easy to quickly generate network diagrams (assuming you had access to a computer with Microsoft Excel). However, it was difficult to customize some aspects of the diagram and was difficult to recover from mistakes.

Strengths
NodeXL has many useful features (such as data importing and autofilling) that made it much easier to quickly generate interesting network diagrams. Data importing made it easy to provide a spreadsheet of edges and automatically extract information about the vertices. In particular, autofill was an extremely useful tool that allowed different encodings to be quickly applied and compared. In addition, various layouts made it simple to meaningfully arrange the data. Also, given how NodeXL is built into Excel, it makes it easy to pick up for people who are familiar with Excel.

Weaknesses
Four main weaknesses of NodeXL I encountered were: operating system requirements, mistake recovery, group customization, and group assignment. I usually use either Ubuntu or Mac. So in order to use NodeXL, I needed to install Microsoft Office on my desktop (fortunately I had already installed Windows for Spotfire). NodeXL made it very easy to change the generated network diagram, but unfortunately it was very difficult to correct mistakes. There were several times when I forgot to change the Layout to “None” before using autofill which resulted in me losing several minutes of work which needed to be manually recreated. Having undo functionality would have greatly alleviated this and made the interface much more enjoyable to use. While using the layouts, it was difficult to position groups in the group-in-a-box layout. Since the nodes were draggable, I assumed the boxes they were in would be draggable as well. So, I was very surprised to learn that this functionality was not supported and ended up compromising with a workaround. Lastly, I was very surprised that NodeXL did not support overlapping groups.

Suggestions
The cost of Windows + Microsoft Excel may be prohibitive for some people, so I think porting across platforms would definitely help NodeXL’s adoption. Supporting undo functionality or at least a way to save the current layout of a graph would enable users to feel safer when exploring the features of the tool. I was happy to see (in class) that extensions to the group-in-a-box layout are currently being developed and suggest that they be made part of the core distribution. I think allowing for overlapping groups to some degree (maybe in a Venn Diagram manner) would add greater flexibility and allow NodeXL to be used for a larger domain of datasets.

References
StackOverflow site: http://stackoverflow.com/
StackExchange Data Explorer: http://data.stackexchange.com/stackoverflow/query/new
NodeXL: http://nodexl.codeplex.com/