Collaboration and Conflicts of Wikipedia's edit network

Introduction & Motivation:

Recent advancement in field of technology has fueled information sharing at unprecedented rate. Knowledge sharing platforms like Wikipedia, Quora, Reddit, Yahoo answers have enabled more active collaboration between readers/viewers and editors/authors. With increase in scale, these platforms attracted more collaboration with time. Among all, Wikipedia is considered the largest collection of information with a more structured collaboration between authors. As primary purpose of its existence, Wikipedia community provides effort to impart correct information to be dispersed among its viewers. Although Wikipedia is multilingual, we will mostly concentrate on English Wikipedia which alone contains over 5 million articles. At present Wikipedia has over 18 billion page views with 500 million unique page views each month[1]. However the primary motivation for choosing Wikipedia for our study is not only because of its huge popularity as a source of information, but Wikipedia has a structured way of logging data about edits on articles and makes the revision history available to researchers. Also the fact that Wikipedia is considered to be treated as encyclopedia to disperse knowledge, the care and thought that goes behind a revision to any article holds much more significance compared to a post or tweets on social media. Unlike Wikipedia, social media platforms tend to provide a platform for a casual conversation among its users and very rarely its users go back to view older posts. On the other hand, Wikipedia readers view articles written a decade ago even if its not modified since. The difference lies in the difference of objectives of these two types of platforms.

It is also important to notice that social platforms flourishes on approximately equal contributions from most of users. However, in Wikipedia we have a really skewed distribution among its number of viewers and number of editors. It is a significantly smaller community of editors and it will be interesting to understand the dynamics of behavior among editors in terms of collaboration and conflicts. These qualitative terms is a bit difficult to quantify on a network. But we can devise some empirical measures that gives us some idea about conflicts and collaboration imposed over Wikipedia. As an overview we can treat reverts on a page as a basis for measurement to construct conflict network. However, a revert means an exact reversion to a previous version of an article and sometimes another editor may modify contents instead of an exact revert. This can be happen more often when another editor changes some part of earlier modification but not the whole, thereby avoiding revert. In this cases it makes much more sense to collect information about all edits which is a superset of reverts. We can design collaboration network from this information and try to analyze the network to find structural imbalance in the network. Also it is possible to view collaboration and disagreement network and try to find correlation among them.

Methods:

1. Revert List Extraction:

Wikipedia revision is typically stored in xml format and because of its huge size, takes considerable time to extract useful information. We took help of modules provided at [5] to fetch the information we need.

The process works as following:

The whole set of pages are considered as dictionary and apart from acting as holder for pages, it stores some metadata about Wikipedia. We need to scan through list of pages using a page iterator to seek the pages we wish to study. Then for each page object, we require a revision iterator to scan through all the revisions to find only the revisions in which revert has been made to an earlier revision. We detect revert using hash field(rev_sha1) which will be identical to a previous revisions rev_sha1 if the pages are identical. This allows to avoid extraction whole text for both the pages and comparing word by word to find mismatch. As typically Wikipedia pages are considerably large in size and most of later modifications are comparatively insignificant and contributes little to existing text, this saves us considerable time. This way we get a list of triplets <reverter_rev_id, reverted_rev_id, revertedTo_rev_id>. We simultaneously maintain a different list containing tuples <rev_id, user_id> where user_id does the revision rev_id on that particular page so that we can fetch user_id of the user whenever required. From previously generated triplets and using the list, we generated triplets of the form <reverter_id, reverted_id, revertedTo_id> i.e. all the revision ids are converted to corresponding user ids.

Note that a large number of revert list contains anonymous users, mostly in reverted section. As users who wish to modify a page anonymously are not assigned any user id, their user id place consist of IP addresses from which modification has been done. If one needs to cluster all anonymous users, it is possible to assign all IP address in the list as a single unique value which would collide with any user IDs e.g. -1.

2. **Edit List Extraction:**

The mechanism to generate edit list follows the same procedure as construction of revert list. However, here instead of focusing only on edits, we take a more holistic view and include all edits. We are mostly interested with the degree of contribution of different editors on a particular page. We maintain a list of all pages of English Wikipedia and for each page we collect all the editors and their number of edits on that particular page. So output of this edit list will look like triplets <page_id, user_id, number_of_edits>. As discussed before, not all editors information is available to Wikipedia as some of them chose to be anonymous while editing. These instances are marked with ‘contributor.id = None’ in data. However, in these cases we can actually look into ‘contributor.user_text’ and that will give us the IP address from which the edits have been done. For now, we included these IPs as different edits but it can be merged in case we want to consider it as a single community.

A sample snapshot of how it looks like:
<page_id, user, num_of_edits>

```
925007,62.64.222.200,1
925007,199.247.128.250,1
925007,75.17.157.194,2
925007,2093577,1
925007,7460363,1
925007,75561,1
925007,80.202.136.181,3
```

3. Degree of collaboration among editors:

a. Collaboration Dictionary

A collaboration dictionary is created with each page id as key and value to be set of duplets <user_id, num_of_edits>. To formally present, we assume page_id is p_i and and user_id u_k with corresponding count as n_k . So dictionary D can be expressed as,

$D = U\ p_i$ where p_i is $U\ u_j$ where $n_j > 0$

We can ignore the count here and consider the contributors as equal if he has done at least one edit on the page. But number of counts provides us with much more information and later we can filter out editors with number of edits less than certain threshold in case we want to treat editors as noise and doesn't create meaningful impact on our model. The dictionary created contains around 36 million keys or pageIDs. Note that not all pages are significant here and some are dummy pages or pages not relevant to our studies and it can be easily detected by noticing its number of edits. A lot of pages does not have any edits and it can be easily filtered out.

Without considering weights, it looks like:

```
14,"[4, 750223]"
15,"[273, 750223, 194203, 135]"
18,"[90, 279219, 166, 750223]"
19,"[300, 750223]"
```

b. Calculating Jaccard Coefficient

We parse the dictionary file and for each pair of page_id (<page_id1, page_id2>), we measured the collaboration as Jaccard Coefficient their overlaps. We can calculate it with or without weights. Both approach have their own significances. Ignoring the weights allows us to create a network where authors are considered as equals irrespective of its contribution to a single page. We can calculate as,

$J(p_i , p_j)$ = |Ui intersection Uj| / |Ui union Uj|

where Uk means the users belonging to page pk.

With weights, we can modify it as

$J(p_i , p_j)$ = \sum_k [min (uik, ujk) / max(uik, ujk)]

Pair of pages with no common users will yield J as zero whereas Jaccard index between itself will give a value of one in both the cases. All other pair values will lie between zero and one.

We can select 2 pages from 36 million pages in 100 trillion ways. This is indeed a huge number to deal with. In practise the number is quite short as some pages are not relevant for our study.

4. Collaboration Network

If we consider each page as a node and the Jaccard coefficient as the weight of edges between two nodes, we can construct a collaboration network. In its present form, the network is dense as we have link between each pair of nodes. However, most of the nodes will be skewed towards zero as Wikipedia as a network is quite sparse because of its sheer amount of information in form of articles. We can see the distribution of Jaccard coefficient among all pages and choose a threshold for filtering the edges of the network. This should give us the collaboration network in its refined form that can be much more useful for community detection.

Wikipedia Database:

Wikipedia arranges its information in entity-relationship model. Databases of Wikipedia are dumped at regular interval and can be found at [2]. For our study, we will consider only following tables:

1. Page:

   Page table holds all relevant information related to a page. Every page is uniquely identified by a page_id. A page name is broken into <page_namespace, page_title>. In order to use the join with other tables, one needs to use both the indices for namespace and title together for correct interpretation. While page namespace holds a number ranging from 0-15, page title contains the actual title of the page with spaces replaced by underscore(_). We will mostly concerned with namespace = 0 to fetch main page information.

2. Pagelinks:

   Pagelinks table represents the network information where source page contains the link of destination pages. Following is the description of Pagelinks:

pl_from: page_id of source page
pl_from_namespace: page_namespace of source page
pl_namespace: page_namespace of destination page
pl_title: page_title of destination page
Note that page_id of target pages are missing which needed to be obtained by using join with page table using ‘pl_namespace’ and ‘pl_title’. The join will create tuples <src_page_id, dest_page_id> both of which together will define the primary key of constructed table.

3. Redirect:

   Redirect table contains all the source and destination page metadata similar to pagelinks table fields but only in case of redirections e.g. when keyword “Obama” is searched in Wikipedia, users are automatically redirected to “Barack Obama”. These information can be considered as virtual mapping.

4. Revision:

   Previous three tables contain metadata and much smaller in size and so Wikipedia normally dumps these information in .sql format. On the other hand Revision history metadata is far larger in size(~44 GB compressed and ~5TB uncompressed) and dumped in .xml format. Revision history stores all revision metadata for all the pages. Following fields are relevant to our studies:

   1. rev_id: Revision ID of the page
   2. rev_user: User_id of the user contributing to this revision
   3. rev_sha1: It stores content based hash and is used in our studies to detect reverts done on the page.

Details of all other kinds of table currently maintained by Mediawiki group is provided at [2]

Controversial Wikipedia Pages:

Difficulties:

The primary difficulties we faced because of the huge scale of Wikipedia and sheer amount of data we had handle at each step of our project. Files at each step of computation became too large to store for a long period of time as well as process efficiently within a limited amount of time.

We have taken a distributed approach to parallelize our effort across several computing node at each step of computations.

a. Wikipedia has revision histories divided across 27 data dumps. Each data dump has been taken as an input and and executed parallely to generate the edit and revert list. Assuming we do not have information of single page distributed across multiple data set, the process ran concurrently because there is no dependency among several instances of the processes.
b. Constructing degree of collaboration among editors require a complex approach because of interdependencies. Here we require each page to be compared with each other page. We execute the process independently, each process will calculate the collaboration of a page with every other page available within its own scope. rather we combine all the file to generate a global edit list and design the program so that each process will be given a part of whole edit list ai and the global edit list A = union aj itself. The task is to calculate Jaccard index of all pages within ai with all pages of A. We intend to choose number of subtasks 'n' so that each of the subtasks can complete its execution within 14 days which is the time limit a program can run uninterrupted on Karst. The processes generated network edges logged over 14 Terabytes of storage in uncompressed format and in order of trillions of edges.
c. Clearly no available tool will let us handle this amount of data to generate a network graph and we need to significantly scale down and introduce sparsity in our dense network. To choose a reasonable data driven threshold, it is important to generate a probability distribution of weights of edges. Our work can be extended from here to construct the network and carry out experiments about collaboration and conflicts on a refined version of Wikipedia network derived from Wikipedia revision history.

Limitations:

Wikipedia is a dynamic network in a sense that every single day there are several hundred thousands edits happening on newly created pages as well as existing pages. The approach taken in not scalable in nature as inclusion of several hundred new pages and thousands of modifications will incur a quadratic order of extra computation at each stage. To truly understand the subject of our research, we need to efficiently design our dynamic which will make the dynamics of our network much more apparent.
It is possible to design our network as sliding window approach rather than a cumulative approach. It makes much more sense that since the inception of Wikipedia the collaboration and conflicts should not be of much relevance compared to recent time dynamics. If we slide the window based on time frame when edits are performed, we avoid dealing with this large computations at each stage. Also the results may seem much more meaningful as it would cause appearance of new nodes or edges between already existing pair of nodes as well as disappearance of nodes and edges from our horizon. Thus introducing a new time axis to study Wikipedia network may simplify the nature and dynamics of our study.

2. Editor Reverts Networks:

The concept of constructing Edit Network from a single Wikipedia page has been proposed in [3] where a Wikipedia page p is visualized as a graph (V,E) where each node v <in> V corresponds to an author and an edge e <in> E corresponds to interaction among author reflected through delete, undelete and restore on page p. This concept of page edit network can be extended into editor reverts network as following:

1. Single page editor reverts network(SpERN)

   A page p in Wikipedia can be considered as G_p = (V_p, E_p) where v <in> V_p includes all the authors who are either a reverter and reverted user specified in triplet list. For each occurrence of <reverter, reverted> pair in the list we either introduce an undirected edge or increase already existing edge's weight by one. Note that simply increasing weight by one makes sense here as we are targeting to measure the number of times of occurrence of disagreement at various times. The resulting undirected weighted network shows the disagreement network or to be specific, a single page editor reverts network. SpERN may contain self-loops as users tend to revert themselves sometimes on Wikipedia.

2. Linked pages editor reverts network(LpERN)

   Pages are linked with one another in Wikipedia creating a network of pages on its own. If we consider a controversial page as a source page, following the links from source page, we can obtain level-1 links network of pages. Similarly it is possible to construct level-i links network where minimum number of hops required from source node to the nodes added at i-th level is 'i'. Using the process described above, we can construct SpERN for each of the graph.

   We will try to construct LpERN from set of SpERNs as following:
   Let SpERN(s) and SpERN(l1) be the source page's and level-1 pages' network respectively. For all G(l1) <in> SpERN(l1), LpERN is defined as,
   LpERN(s,l1) =

   Results:

References:

[1] https://en.wikipedia.org/wiki/Wikipedia
[2] https://dumps.wikimedia.org/enwiki/
[2] http://www.mediawiki.org/wiki/Manual:Database_layout
[3] U. Brandes, P. Kenis, J. Lerner, D. van Raaji. Network Analysis of Collaboration Structure in Wikipedia. In WWW, 2009
[5] http://pythonhosted.org/mediawiki-utilities/