–Independent Work Report Spring,2014–

# Exploring Self-Organizing Learning Groups in an Online Learning Platform: Classroom Salon

Brandon Lam
Advisor: Dr. Ananda Gunawardena

May 6, 2014

## Abstract

*Collaboration in a physical classroom is much different than collaboration that appears on an online platform. Classroom Salon is an online learning platform that encourages students to work together and learn from one another through its interactive features. However it is harder to identify the groups that from as a result of this type of collaboration than those formed in a classroom setting. I will use data from Classroom Salon to identify the groups that are created via the platform. These groups are called self-organizing learning groups since they are formed from the interactions between users rather than from external influences that can either restrict or facilitate the formation of particular groups. I explore three different approaches to identify and evaluate groups that form over Salon 1906, a sophomore literature class. The three avenues that I explore are structure, content, and rhetoric. Through the use of Classroom Salon, WordNet, and Docuscope, I find that due to the nature of each of these methods, the groups found by each are very different.*

## 1. Introduction

Looking at self-organizing groups can be very useful; it can provide insight into the success and level of collaboration between individuals. This is of particular importance when dealing with online platforms. In a physical classroom, such as Mitra's Schools in the Cloud, the effectiveness of collaboration is evident. A teacher can visually see the groups that tend to from, monitor the content that is discussed among groups, and recognize the emotional impact and types of relationships that form within the groups. However this is not as readily evident in a online environment. Also, by looking at other factors, I can gain insight into how these groups are formed. My goal of this project is to better understand the collaboration and connectivity between students over an online platform by analyzing the self-organizing learning groups that from via Classroom Salon.

Classroom Salon is an online learning platform that is used by professors to teach online through a collaborative environment. It provides an enviornment through which students can learn, collaborate, and discuss the topics learned in class. Salon 1906 is a sophomore literature class taught by a professor at a college in Virginia [10]. I use Salon 1906 as the source of data my research. In my research I explore methods in finding self-organizing groups. In particular I look at groups in the following ways:

- Structural Analysis: I have developed a way to take the comments, create a networking graph based on students responding and interacting with other students, and use a clique-finding algorithm to find the learning groups that have formed structurally. This takes advantage of Classroom Salon's design and utilizes the many discussion threads to form this connected graph.
- Content Analysis: I have incorporated WordNet to take the comments that each indivdiual has made and categorize each individual into a number of categories. From this categorization, I find the groups that have formed based on the content of the students' comments. This approach attempts to establish relationships between individuals who tend to discuss similar topics.
- Rhetorical Analysis: I have incoporated the use of a corpus-based rhetorical analysis tool called Docuscope in order to provide the analysis needed to group indiduals based on their rhetorical tendencies.

## 2. Context

In a traditional classroom setting students situate themselves in front of a teacher, who gives lectures and guides students through the reading of various texts. However, this form of education "assumes that kids are empty vessels who need to be sat down in a room and filled with curricular content" [13]. Sugata Mitra, an educational researcher and professor at Newcastle University, argues that this way of teaching is wrong and unnecessary [4]. Through his "Hole in the Wall" experiments, Mitra has shown that curisosity and peer interest can drive children to teach themselves and each other, even in the absense of supervision and direct influence from a teacher. What makes this possibe is the evolution of the Internet and the ability for childern to explore their curiosity by going online to access the plethora of resources available to them [4].

As part of a series of the "Hole in the Wall" experiments conducted by Sugata Mitra, Mitra and his colleagues dug a whole in the wall of an urban slum of New Delhi in 1999. They installed an Internet-connected computer in the wall and left it there. With a hidden camera monitoring the computer, Mitra discovered that within a day childern, who had no prior experience with computers, started to play with the it and in the process learned how to go online. He also found that childeren were motivated to teach each other how to use the computer [1]. Mitra's experiments suggest that children have an inherent curiosity that has the potential drive collaborative learning even in the absense of supervision or teacher influence.

In 2013 Sugata Mitra's work led him to win the 2013 TED Prize, wishing to create a School in the Cloud. A School in the Cloud is a learning lab that provides an environment through which children can embak on intellectual endeavors by utilizing the resources of information online. The School in the Cloud promotes self-organizing learning environments (SOLE) by controlling the influence made by adults. It encourages collaborative efforts and allows students to learn more through their intelectual curiosity. However, Mitra does not, however, discredit the value of the influence of teachers and mentors. He believes that as the availability of resources via the Internet increases, the need for a traditional classroom environment decreases since children can teach themselves. Mitra suggests that the role of the the teacher should pivot from being the source of knowledge, as it typically is in the classical setings, to being the mentor who facilitates the growh of the children's curiosity and promotes the exploration of this curiosity. In parallel with trying to create Schools in the Cloud, Mitra urges parents and teachers to start their own SOLEs while embracing the principles of a School in in the Cloud [3, 5].

## 3. Background

### 3.1. Classroom Salon

Ananda Gunawardena, Former Associate Teaching Professor of Computer Science at Carnegie Mellon University, and David Kaufer, Professor of English at Carnegie Mellon University, through the use of thier online learning platform, Classroom Salon, seek to build on Sugata Mitra's focus on collaborative learning. Classroom Salon puts great focus on interactive content. The platform not only assists the professors and gives them insight about their classes through data-driven analytics, it also promotes self-learning to the students through its collaborative design. The idea behind Classroom Salon is that it is a place where students can gather and talk about different ideas and topics. A professor will upload documents and videos to a salon and can choose to provide prompts or other instructions for the students. The files are then accessed by the students, who can view the annotations of other students and annotate the documents themselves with comments, questions, or replies to other another post. As a result of this design, Classroom Salon allows professors to encourage collaborative interactions and individual curiosity while being able to monitor, view, and analyze the learning process of their students. [2].

### 3.2. WordNet

WordNet is a large English lexical database managed by members of Princeton University and supported by the National Science Foundation under Grant Number 0855157. It is able to groups nouns into sets of cognitive synonyms, called synsets. WordNet also uses the super-supordinate relation, or hyperonymy, in order to represent the hierarchies of nouns. For example, in the WordNet hierarchy, the word "fruit" will be higher in the hierarchical structure than the word "apple" since an apple is a type of fruit. The closer a word is to another in the WordNet network, the more similar the two words. WordNet also incorporates meronymy in its database. Meronymy is the part-whole relationships between synsets. For example, Wordnet recognizes that a "leg" can describe a part of a "chair." Due to the hyponymy, WordNet then can deduce that a "leg" can describe a feature of an "armchair" [15]

### 3.3. DocuScope

DocuScope is a piece of software, started in 1998 by David Kaufer and Sugugru Ishizaki of Carnegie Mellon University, that can anaylze texts through a variety of interactive visualization tools and can provide metrics to evaluate different texts in a number of fields. In addition to being able to do advanced rhetorical analysis, DocuScope is also effective in developing a dictionary in a systematic fashion. As a result, analysts can use DocuScope with domain-specific dictionaries. The rhetorical analysis of DocuScope can provide information regarding the levels of emotion, character, interaction, and a variety of other fields. [9].

## 4. Methods and Techniques

### 4.1. Structural Analysis

The first approach to finding the self-organizing groups that exist in Classroom Salon is the structural approach. Classroom Salon's interactive features provide an easy way for students to comment and reply to one another. A structural representation of the interactions that have occured in the salon is a logical attribute to analyze. The Classroom Salon platfrom gives the structural representations in the form of threads. Each thread represents a set of comments and replies that all extend from one initial annotation and contains many sub-threads. Each thread can be represented as a tree. I take these threads and merge them in order to create a network that represents the connections within the whole salon.

*Creating a Networking Graph:* In order to create a network of the students, I took all of the comments and observed who responded to whom. In a simple representation there would be an edge between person A and person B if for at least one occurance person A replied to person B or person B replied to person A. However given the whole salon and all the documents within it, there is a high probability that there would be an edge between given any two vertices. As a result, this representation would lead to a graph with many edges and a very few (if not zero) number of single nodes. For instance, Figure 1 demonstrates this (graphed by an open-source program called yEd Graph Editor and utilizing a external java library, jxl.jar, to export data into an excel file) [6, 8]

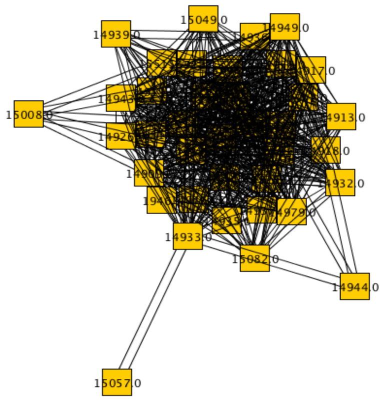
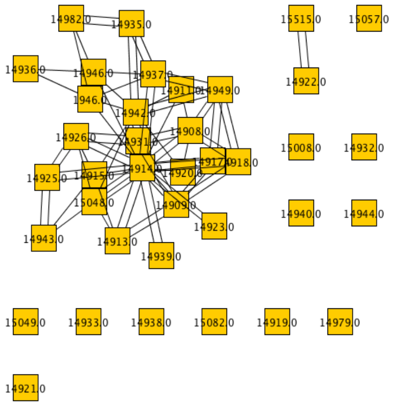


**Figure 1: Salon 1906 - Graph with a threshold of one interaction per edge**

As seen in this graph, there are no single nodes and there is a great number of edges that connect the nodes in the center of the graph. In order to reduce the noise of the data, I revised the definition of an edge to be that there exists an edge between student A and student B if there are at least four interactions between students A and B. Since I am trying to find self-organized learning groups, multiple interactions are necessary since two people should not be considered to be in the same group if they have only minimal interaction. Each pair of individuals inf a given group whould have recurring interactions. The result of this more specific definition of an edge is shown below in Figure 2 (using yEd) [6, 8]:



**Structural Network: Threshold - 4**

It can be noted that this graph contains far fewer edges and is much more readable. The single vertices also show which members of the salon did not interact with any partiular group of people. Each edge is more significant, which can lead to the finding of more strongly connected groups.

*Bron-Kerbosch Algorithm:* One approach to define what criteria is required in order for there to ge a group is to require that in order for any person to be considered part of a particular group, the individual had to form a clique with the other members of the group *clique*. In graph theory, a clique is a set of vertices in an undirected graph, such that each vertex in a particular set of vertices is connected by an edge to every other element in the set [11]. A way to solve this clique problem is to implement the *Bron-Kerbosch algorithm*, which is an algorithm that finds maximal cliques in an undirected graph. A maximal clique is a clique that is not a subset of another clique. The way the algorithm works is that given three sets *R, P,* and *X*, find the maximal cliques that include all the nodes in *R*, some of the nodes in *P*, and none of the nodes in *X*. In each call of the algorithm $P \cup X$ is the set of vertices that are connected to every element of R. Therefore when *P* and *X* are empty sets, there are not more vertices that can be added to *r*, so *R* is a maximal clique. This backward recursive algorithm is initiated with both *R* and *X* as empty sets and *P* as the vertext set of the graph [7]. The pseudocode is described below in Figure 3 [7]:

```
BronKerbosch (R, P,  X) {
        if (P and X are both empty)
                { return R as a maximal clique }
        for each vertex v in P {
                BronKerbosch(R ∪ {v}, P ∩ N(v), X ∩ N(v))
                P := P ∩ {v}
                X := X ∪ {v}
        }
}
```

**Figure 3: Pseudocode for the Bron-Kerbosch algorithm**

## 4.2. WordNet Analysis

The second approach to grouping the students into clusters involves analyzing the content and grouping students based on what topics they referred to in their annotations and replies. A connection does not necessarily mean that two students interacted with one another. People can also be connected by the things that they say. For instance, if two people tend to talk about a particular subject but do not directly show this similarity through the structure of the platfrom, they can still be connected. They may not respond to each other's posts, but they may read each others comments, have similar views, or may want to increase interaction in the future. In order to determine the connectivity between individuals, I look at which topics each individual tends to make comments on and compare these topics. I then create a graph, defining an edge if two people have similar preferences in the topics that they choose to write about. Running the Bron-Kerobsch clique-finding algorithm again, some of the groups that form are evident.

*Bucket Finding:* For any given salon, the number of different topics can vary from one big topic to many more specific topics. Therefore, in order to categorize the users into buckets based on content, we must first determine which buckets are significant. To do so, I use a WordNet of nouns. Nouns are relatively easy to analyze and the relationships between them can be easily defined.

In order to find these buckets, I developed an algorithm that utilizes WordNet and SAP, or shortest ancestral path. The shortest ancestral path between two words returns the common ancestor that lies in the shortest path between two words. SAP takes in a directed graph for the constructor while using a breadth-first-search algorithm for its methods. I modified Digraph.java and BreadthFirstDirectedPaths.java of Robert Sedgewick and Kevin Wayne to suit my puroposes [14, 12]. The pseudocode of the shortest ancestral path is as follows in Figure 4 [16]:

```
Sap(A, B) {
        Run Breadth-first-search on both A and B to find
                nodes that A and B can reach by a path.
        Get distance from A and B to every other node
        Use these distances to find shortest distance from A to B
                keep track of common ancestor
        return common ancestor;
}
```

**Figure 4: Shortest ancestral path method**

The general idea of the algorihm I used to find the bins is described below. In the pseudocode *A* represents the set of word to be categorized and *B* represents the set of buckets, in which the words shall be categorized.

```
GetBins(A) {
        initialize B;
        create a WordNet wn;

        // Get initial set of words for B
        for (each word in A) {
                for (each word2 in A) {
                        String ancestor = sap(word, word2);
                        set to be closest ancestor if necessary;
                }
                add closest parent to B;
        }

        // Remove words that are close to top of WordNet
        for (each word in B) {
                if (wn.distance(word, "entity") <=  threshold)
                        {remove word from B;}
        }

        // Remove if a more specific version of another word
        for (each word in B) {
                for (each word2 in B) {
                        if (wn.sap(word, word2) == word2)
                                {remove word form B;}
                        if (wn.sap(word, word2 == word)
                                {remove word2 from B;}
                }
        }
        return B;
}
```

**Figure 5: Algorithm to find the buckets that will classify the annotations**

In order to find words that are similar to two words, this algorithm indirectly uses BreadthFirst-Search in order to find the shortest ancestral path between the two words. For each word, the algorithm adds the closest ancestor that was calculated in the SAP findings. After doing this for each word, the set B contains a set of elements that could be possible buckets of the original words. The algorithm then removes any bucket that is too close to "entity" (the top of the hierarchial structure of the WordNet). It does so by calculating the distance between "entity" and each of the buckets. In this case, a bucket was removed if the shortest path between it and "entity" was less than or equal to five. The algorithm then removes buckets that are too specific, meaning that a bucket is a descendant of another bucket. It uses SAP to find these elements and then removes them. This algorithm can be slightly modified into a recursive algorithm in order to reduce the number of buckets generated. However, I did not find it necessary for Salon 1906, for the number of buckets was reasonable. From roughly 2,500 nouns, I narrowed the number of buckets down to roughly 25 buckets.

*Categorizing Users:* After finding which topics existed among the annotations in Salon 1906, I categorized the users into the different buckets. To do so, I aggregated all of the comments that each a particular user made (keeping only the nouns). Then for each user I took their set of nouns and placed each noun into a particular bucket, or multiple buckets, if there was a directed path from the bucket name to the noun. After putting each noun into the necessary buckets (or no bucket if not significantly related to any particular bucket), I created weights relating the probabilities of each person making a comment about a particular topic to its corresponding bucket. For example, if the nouns that a person used were placed into a particular bucket .1% of the time, the weight of that person being in that bucket would be .1%.

*Creating a Graph and Finding Clusters:* Similarly to creating a graph based off the strucural analysis of the data, when dealing with contexual analysis there also needs to be a threshold that determines if there is an edge or not between two individuals. In this case we must set two threshold points.

The first theshold point, which I will call the weight-threshold, is the weight that is required. For example, say a person is categorized with weights 0.01, 0.01, and 0.001 into buckets A, B, and C, respectively. In this case, person A may have referred to category C by accident, which would account for the lower weight. If we set the weight-threshold to be 0.0011, we can disregard the last bucket for this person. By setting a weight-threshold we can avoid random chance and less relavent buckets for particular individuals.

The second threshold point, which I will call the bucket-threshold, is what really determines if two people are connected, meaning they have an edge between them in the graph. We do this by setting the number of shared buckets that is required. For example, if we set the bucket-threshold to be 3 buckets, any two people who are in at least three common buckets are considered to be connected. Say there are three students (A, B, C). Student A fits in buckets 1, 2, and 3. Student B fits in buckets, 2, 3, 4, 6. Student C fits in buckets 3, 4, 5, and 6. We have already included the weight-threshold mentioned above. In this case, the only edges in this subset of three nodes is the one between students B and C since they are both placed in buckets 3, 4 and 6.

### 4.3. Docuscope Analysis

The final approach that I use in this paper is to use rhetorical analysis as the tool recognize the groups that form. This approach incoporates the use of DocuScope, a text analysis environment that is able to provide a variety of visual tools along with metrics that describe many different dimensions. Unfortunately, DocuScope cannot be run outside the DocuScope research group and its studnets. [9].

Since DocuScope is already an implemented software the preparation before using the software was minimal. I submitted a set of text documents to be analyzed. Each text document contains the concatenation of all of the comments made by a particular individual. By submitting the documents to DocuScope, the program outputted a series of metrics for a variety of dimensions including levels of subjectivity, emotion, assertiveness, etc.

For simplicity and the fact that DocuScope can be run only by its reasearch team, I only ran DocuScope on seven different text files. The students who make up these text files are mixed throughout the distance learning section and the face-to-face section. They are also spread throughout the different groups that the professor created.

### 4.4. Human-Encoded Data

As part of the course, the professor of Salon 1906 assigned students in his class into a number of groups. The groups made, with the students' user ids, were as follows:

- Group 1: 14979, 14918, 14946, 14937, 14914
- Group 2: 15515, 15057, 14922, 14913, 14925, 14936, 14931
- Group 3: 14949, 14942, 14921, 14911, 14917, 14915
- Group 4: 14908, 14939, 14926, 15049, 14933, 14944
- Group 5: 14923, 14909, 14982, 14943, 14920, 14938, 14940
- Group 6: 14919, 14932, 15048, 15008, 14935, 15082

(Note: Groups 1 - 3 are the distance learning sections; Groups 4 - 6 are the face-to-face sections) [10]

## 5. Interpretation of Results

### 5.1. Structural

After running the Bron-Kerbosch algorithm on the graph formed with salon 1906 data, there were a total of 44 cliques. The size of the cliques ranged from a single node to three members in a clique. From my results, I found that 18 students were only part of cliques of a single vertex, which is consistent with the fact there are 13 individulas in the graph that did not interact frequently enough with any other person. There were 21 cliques of size 2 and 10 cliques of size 3. By looking at the cliques that formed, student 14914 seemed to provide the most influence for she participated in 9 out of the 10 size-3 cliques.

Out of the 21 size-2 cliques, 10 were composed of students form the distance learning sections. Out of these 10 cliques, five of the cliques were of students from the same group. In constrast, only one of the cliques of size 2 was composed strictly of mebmers of the face-to-face session. Overall it seems as though the online participants were more interactive and collaborated more with the other students. They not only responded to other online members, they also interacted with the face-to-face students. In fact, there was at least one member from the distance learning section in all but one of the cliques of size 2 or more. Perhaps this is due to the fact that face-to-face students had a greater opportunity to collaborate during class, outside the online platform. For the distance learning section, Classrom Salon as their main instrument of communication. This may explain the greater frequency in participation from the online students, causing them to create more online connections with people. The groups found using this method provides useful insight into the value of online learning and the ability for student to learn in a collaborative, curiosity-driven setting. It shows that even in minimal supervision and influence from a teacher, and the lack of a physical relationship between members, students were still motivated to participate in discussion and form their own sets of groups.

It is also evident that the assigned groups that the professor created did not have a significant effect on the formation of self-organizing groups via Classroom Salon. Using the criteria for identifying cliques, there were only six cliques that consisted soley of members from the same group. Four of these cliques involved student 14942, which suggests that these cliques from a single group was not due to the sociability of the group, but rather the sociability of a specific member. The other cliques found were either single nodes or consisted of students from a variety of groups. The fact that the external groups had little effect on the groups found through structural analysis is reasonable because students were not required to respond only to their group members.

## 5.2. Contextual

In the case of Salon 1906, I set the weight-threshold to be 0.005 and the bucket-threshold to be 6. After running the series of algorithms with these thresholds, the data produced 11 different cliques. Only one of these cliques consisted of a single node. Only student 15057 was in her own group. The other 10 cliques remaining consisted of members from different groups and of both the distance and face-to-face groups. Comparing this with the findings from the structural analysis, content analysis provides a fewer number of cliques, but on average each of these cliques has a greater number of students. The cliques that are found by using this analysis are reasonable given the design of Classroom Salon. For the most part, any given document that is posted has a finite number of topics that it mentions. Therefore if two different people annotate and comment the same document, there is a relatively high probability that both individuals refer to the same topic. This increases the number of people that fit in each bucket and explains the fact that there is a greater number of students in each clique. This also indirectly decreases the number of cliques since the probability of the occurance of a clique that contains a single node also decreases.

However, the structural and contextual analyses do not only show contrary data. For example, the one student in her own clique from the contextual anlaysis, student 15057, was also in a clique of a single node when using a structural anaysis. This is a logical occurance because if a student tends to discuss topics contrary to the common topics of the others, she is less likely to interact with them.

## 5.3. Rhetorical Analysis

One way to compare the students through the DocuScope analysis is by analyzing the differences between the different dimension metrics. For example student 14946 was rated with a 6.21 in the Personal Register field, while student 14921 was rated with a 6.39. Since the difference between these two numbers is relatively small, it is possible to conclude that these students are similar in regards to their Personal Register, which measures the subjectivity of their comments. I compare the students within the salon with regards to two factors: the sum of the cluster metrics and the cluster "interactive." A subset of the results from DocuScope are shown below:

| **Student ID** | **14946** | **15048** | **14942** | **14979** | **14921** | **14926** | **14931** |
|---|---|---|---|---|---|---|---|
| Interactive | 1 | 1.09 | 1.04 | 2.31 | 2.5 | 1.03 | 1.32 |
| Total | 200.53 | 202.35 | 195.93 | 189.7 | 182.99 | 204.09 | 212.23 |

**Table 1: DocuScope results for Interactive and Total clusters**

The data shown above suggests that students 14946 and 15048 are highly similar overall since their overall ratings are 200.53 and 202.35 repsectively. Similarly students 15048 and 14926 are highly related. We can expand this group to include student 14942. Students 14979 and 14921 can also be put in a group using this metric.

Similar results are shown when observing the similarities in regards to the *interactive* rating. Students 14946, 15048, 14942, and 14926 appear to be very similar in their interactive ratings. In both metrics student 14931 seems to be in her own group. However, there does not seem to be significant influence of the groups created since the members of the groups visible through the docuscope analysis come from different assigned groups. This is consistent with the data found from the other approaches of analysis.

However, DocuScope suggests that Students 14979 and 14921 are the most influential in this group of seven since they both have significantly greater interactive ratings than the others. The content analysis does not provide much insight into this, but the structural analysis seems to provide contrary evidence. When viewed in a structural context, both of these individuals, 14979 and 14921, fail to establish a significant connection with any other student. On the otherhand, student 14946 and 14926 received the lowest interactive ratings. However they participated in four and three cliques, respectively, from the structural analysis. The contextual analysis suggests that they were all part of the same group. These findings from the use of docuscope and the structural analysis suggest that the presense of structural interaction may not necessarily suggest rhetorical interaction and vice-versa. However, this may be explained by further exploration into who tends to reply to others, for a high interaction rating may be related to the frequency of replies since an annotation at the start of a thread may be less likely to exhibit interaction cues.

## 6. Conclusions and Future Works

### 6.1. Conclusions

In conclusion, there are a variety of ways to analyze the formation of groups through an online learning platform. In the study of Salon 1906, a sophomore literature class, through the use of the learning platform, Classroom Salon, I analyzed the formation of self-organizing learning groups using three different approaches: structural, contextual, emotional and rhetorical.

The first approach of structural analysis was supported by the internal design of Classroom Salon, which monitors the interactions between users through the representation of different threads. Upon implementing a maximal clique-finding algorithm, the Bron-Kerbosch algorithm, the existance of many cliques, each being small in size, became evident. The cliques found suggest that the students that made up the distance learning section were more inclined to interact with other users, not only with other online users but also with members of the face-to-face section, on a more consistent basis. The face-to-face members, although they interacted less frequently on average, showed a similar behavior in interacting with both sets of students.

The second apporach in using contextual analysis to determine the self-organizing learning groups provided another perspective on the problem. After determining the major topics discussed and categorizing the users into these topics, the analysis provided evidence contrary to the structural analysis. Instead of finding a multitude of small self-organizing learning groups, the contextual analysis, which also took advantage of the Bron-Kerbosch maximal clique-finding algorithm, suggests that there are only a few groups that form, each of which contains many members.

DocuScope provides great analysis into how rhetorical analysis can determine what groups are forming. The metrics used in DocuScope can provide useful insights into how similar individuals are in their rhetoric style. Classifying in this way can help identify groups that may work well together. The analysis also provides insight into how interactive certain individuals are. From comparing with the structural analysis, the data suggests that there is not necessarily a correlation between the interactiveness of an individual and the number of cliques the individual is a part of structurally.

Throughout all three approaches in identifying self-organizing groups the results vary drastically. However all three show evidence that the online platform is much different than the physical platform. Although the members of this class were assigned to be in specific groups, there seems to be no support for the hypothesis that the formation of online groups is dependent on the physical groups.

### 6.2. Future Works

Looking at self-organizing groups, espcially through online learning platforms, is a relatively new idea influenced by Sugata Mitra's work. As a result, there are many directions in which research can be focused. For the structural analysis, it is possible to define a group in a different way. A group does not necessarily have to be defined as a clique in a graph. Clustering, based on distances between nodes, is just one of the many directions one can take. Clustering would then allow two students to be part of the same group even if they were not connected by an edge; instead they would be connected via another node in the graph. Also the definition of an edge can also change. Edges can be directed, have weights, or require a different threshold. Each of these approaches would require a different way of analysis, but are all valid options.

For the WordNet approach, one might pick which buckets exist by utilizing another algorithm, which may be more based on WordNet hierarchy rather than shortest ancestral paths. The weight- and bucket- thresholds could also be experimented with to further analyze the groups. Furthermore, analysis could extend to incorporate more elements of WordNet. This includes verbs, adjectives, domains, etc. [15]

There is also much that can be expanded on when dealing with the DocuScope analysis. For instance there are a lot of different levels of specificity to the DocuScope metrics, ranging from *clusters*, like "Emotion," to specific *classes*, such as "Emotion_Anger" [9]. One might also want to do a similar anaysis as I did with WordNet. After retrieving data analysis from the DocuScope program, it is possible to use this information to create a graph and run a clique-finding algorithm or another algorithm to find the groups. In this case, one might define an edge between to students if the difference between a specific metric or groups of metrics is smaller than a certain threshold. One may also require that in order for two users to be connected by an edge, they must also share a specific number of metrics that satisfy this threshold.

One might also continue analysis in order to evaluate the effectiveness of the groups found. I looked at ways to group specific users together, however there is no metric to value the collaboration between these members. With further analysis of self-organizing learning groups, Classroom Salon and other online learning platforms, may be able to implement features that continue to link self-organizing learning groups and further encourage collaboration between members.

## 7. Acknowledgements

My work was greatly benefited by the thoughts, advice, and support of Dr. Ananda Gunawardena, Professor Adam Finkelstein, Dr. Christiane Fellbaum, Professor Miles Mccrimmon, and Professor David Kaufer. They helped guide me through the research process and provided me with their expertise regarding Classroom Salon, WordNet, and Docuscope. My work would not have been possible without the help of each and every one of these individuals.

## References

[1] (2007) Kids can teach themselves. video. [Online]. Available: http://www.ted.com/talks/sugata_mitra_shows_how_kids_teach_themselves
[2] (2013) Classroom salon. [Online]. Available: http://www.classroomsalon.com
[3] (2013) School in the cloud - sugata mitra. [Online]. Available: http://www.ted.com/participate/ted-prize/prize-winning-wishes/school-in-the-cloud-sugata-mitra
[4] (2013) Sugata mitra. [Online]. Available: http://www.ted.com/speakers/sugata_mitra

[5] (2014) School in the cloud. [Online]. Available: https://www.theschoolinthecloud.org
[6] (2014) yed graph editor. [Online]. Available: http://www.yworks.com/en/products_yed_about.html
[7] A. Conte. (2013, May) Review of the bron-kerbosch algorithm and variations. Available: http://www.dcs.gla.ac.uk/~pat/jchoco/clique/enumeration/report.pdf
[8] E. Jung and D. Lambert. (2012, Oct.) A java library for reading/writing excel. Available: http://sourceforge.net/projects/jexcelapi/
[9] D. Kaufer and S. Ishizaki. Docuscope: Comupter-aided rhetorical analysis. Available: http://www.cmu.edu/hss/english/research/docuscope.html
[10] M. Mccrimmon, "Salon 1906 info," personal communication, Apr. 2014.
[11] J. Moon and L. Moser, "On cliques in graphs," *Israel Journal of Mathematics*, vol. 3, no. 1, pp. 23–28, Mar. 1965.
[12] R.Sedgewick and K. Wayne. (2013, Dec.) Digraph.java. Available: http://algs4.cs.princeton.edu/42directed/Digraph.java.html
[13] D. Searls, "Natural forces," *Linux Journal*, vol. 95, Mar. 2002.
[14] R. Sedgewick and K. Wayne. (2013, Oct.) Breadthfirstdirectedpaths.java. Available: http://algs4.cs.princeton.edu/42directed/BreadthFirstDirectedPaths.java.html
[15] P. University. (2010) About wordnet. Available: http://wordnet.princeton.edu
[16] A. Yabroudi and B. Lam, "Sap algorithm," coding assignment, 2012.

"I pledge my honor that I have not violated the honor code during the writing of this paper."

– Brandon Lam

## 8. Appendix:

**Exhibit 1: Comment data structure**

```
public class Comment {
        private String name, email, document, annotationId, documentId,
                userId, commentText, commentArea, commentPositivity,
                commentTime, replyTo, status, modifiedTime, isAnonymous,
                annotationParagraphId, annotationId1, paragraphId,
                startChar, endChar, annotationText, salonId;
        ...
}
```

**Exhibit 2: Bron-Kerbosch**

```
public class BronKerbosch{
        ...
        private void runAlgorithm(HashSet<String> r, HashSet<String> p,
                HashSet<String> x) {
                if (p.isEmpty() && x.isEmpty()) {
                        for (String elem: r)
                                System.out.print(elem + ",␣");
                        System.out.println();
                }

                HashSet<String> pclone = new HashSet<String>(p);
                for (String vertex: p) {
                        HashSet<String> r2 = new HashSet<String>(r);
                        r2.add(vertex);
                        HashSet<String> p2 =
                           getIntersect(pclone, neighbors.get(vertex));
                        HashSet<String> x2 =
                           getIntersect(x, neighbors.get(vertex));

                        runAlgorithm(r2, p2, x2);
                        pclone.remove(vertex);
                        x.add(vertex);
        }
        ...
}
```

**Exhibit 3: SAP**

```
public class SAP{
        ...
        // a common ancestor of v and w that participates in a shortest
        // ancestral path; -1 if no such path
        public int ancestor(int v, int w) {
                if (v < 0 || w < 0 || v > graph.V() - 1 || w
                        > graph.V() - 1)
                        throw new
                          IndexOutOfBoundsException("inputs␣not␣valid");
                length(v, w);
                return comAnc;
        }
        ...
}
```

**Exhibit 4: FindBins - first part of finding buckets**

```
public class FindBins{
        ...
        public HashSet<String> getBins(HashSet<String> words) {
                // Find Possible Bins
                HashSet<String> bins = new HashSet<String>();
                for (String elem : words) {
                        double minDist = Double.POSITIVE_INFINITY;
                        String index = "";
                        int dist = 0;
                        for (String elem2 : words) {
                                if (elem2.equals(elem))
                                        continue;
                                if (!wn.isNoun(elem) ||
                                        !wn.isNoun(elem2))
                                        continue;
                                if ((dist = wn.distance(elem, elem2))
                                        < minDist) {

                                        index = elem2;
                                        minDist = dist;
                                }
                        }
                        if (!wn.isNoun(elem) || !wn.isNoun(index))
                                continue;
                        bins.add(wn.sap(elem, index));
                }
                return bins;
        }
        ...
}
```

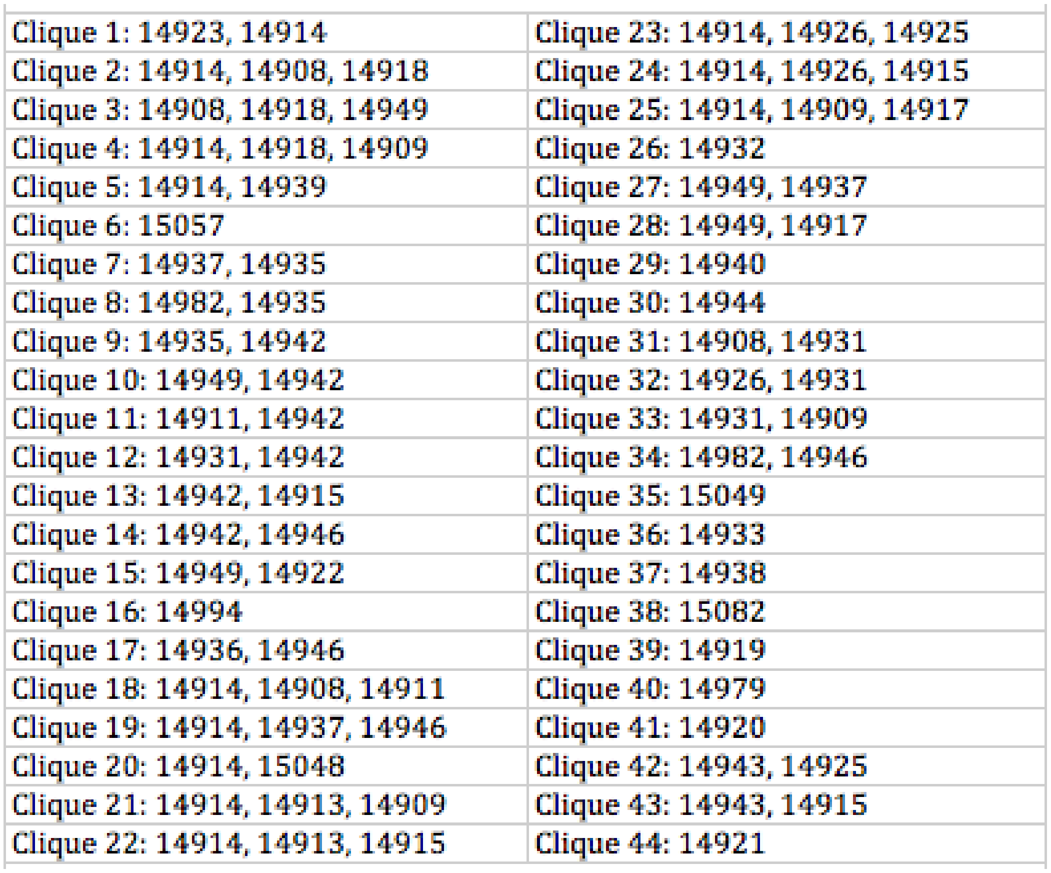
**Exhibit 5: ReduceBins - second part of finding buckets**

```
public class ReduceBins {
        public static void main(String[] args) throws IOException {
                String line;
                BufferedReader reader =
                        new BufferedReader(new FileReader(args[0]));

                HashSet<String> bins = new HashSet<String>();
                while ((line = reader.readLine()) != null) {
                        bins.add(line);
                }

                WordNet wn =
                        new WordNet("synsets.txt", "hypernyms.txt");
                HashSet<String> removalElems = new HashSet<String>();
                for (String elem : bins) {
                        if (wn.distance(elem, "entity") <= 5) {
                                removalElems.add(elem);
                                continue;
                        }
                        for (String elem2 : bins) {
                                if (elem.equals(elem2))
                                        continue;
                                if (wn.sap(elem, elem2).equals(
                                                elem)) {
                                        removalElems.add(elem2);
                                }

                                else if
                                        (wn.sap(elem, elem2).equals(
                                                elem2)) {
                                        removalElems.add(elem);
                                }
                        }
                }
                for (String elem : removalElems) {
                        bins.remove(elem);
                }
                for (String elem : bins) {
                        System.out.println(elem);
                }
        }
}
```

**Exhibit 6: Eliminate Children - third part of finding buckets**

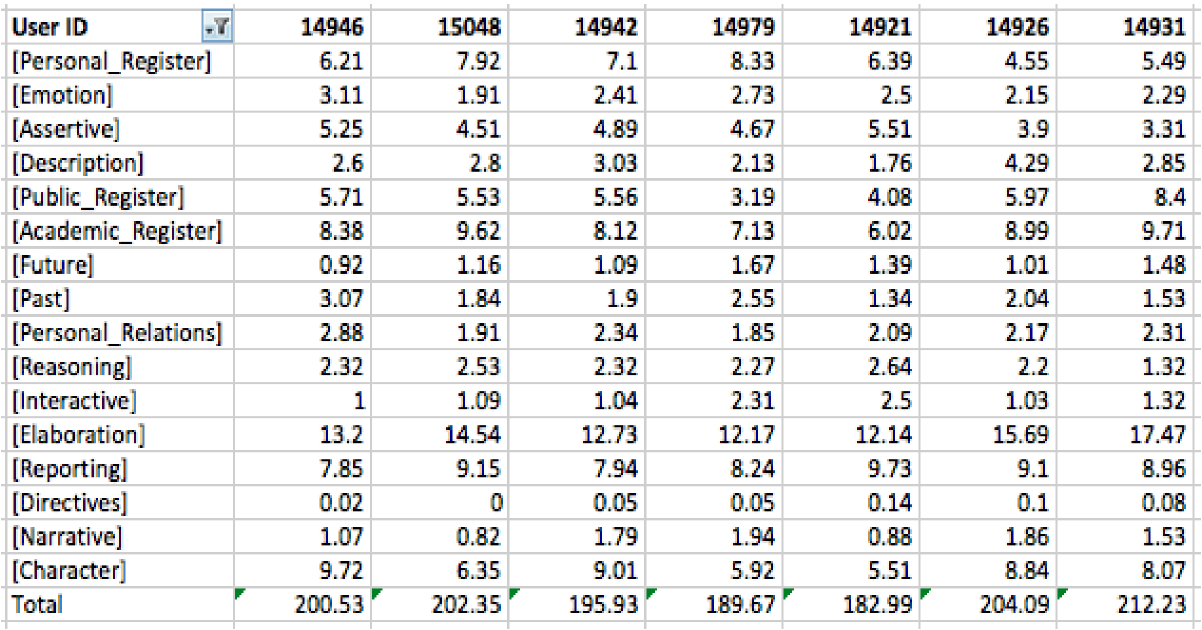
```
public class EliminateChildren {
        public static void main(String[] args) throws IOException {
                String line;
                BufferedReader reader = new BufferedReader(
                        new FileReader(args[0]));

                HashSet<String> bins = new HashSet<String>();
                // Read file to store all the comments
                while ((line = reader.readLine()) != null) {
                        bins.add(line);
                }

                // String[] check = bins.toArray(
                        new String[bins.size()]);

                WordNet wn =
                        new WordNet("synsets.txt", "hypernyms.txt");
                HashSet<String> removalElems = new HashSet<String>();
                for (String elem : bins) {
                        for (String elem2 : bins) {
                                if (elem.equals(elem2))
                                        continue;
                                if (wn.sap(elem, elem2).equals(elem)) {
                                        removalElems.add(elem2);
                                }

                                else if (wn.sap(elem, elem2).equals(
                                                elem2)) {
                                        removalElems.add(elem);
                                }
                        }
                        }
                for (String elem : removalElems) {
                        bins.remove(elem);
                        }
                for (String elem : bins) {
                        System.out.println(elem);
                }
        }
}
```

**Exhibit 7: Structural Analysis - Cliques**



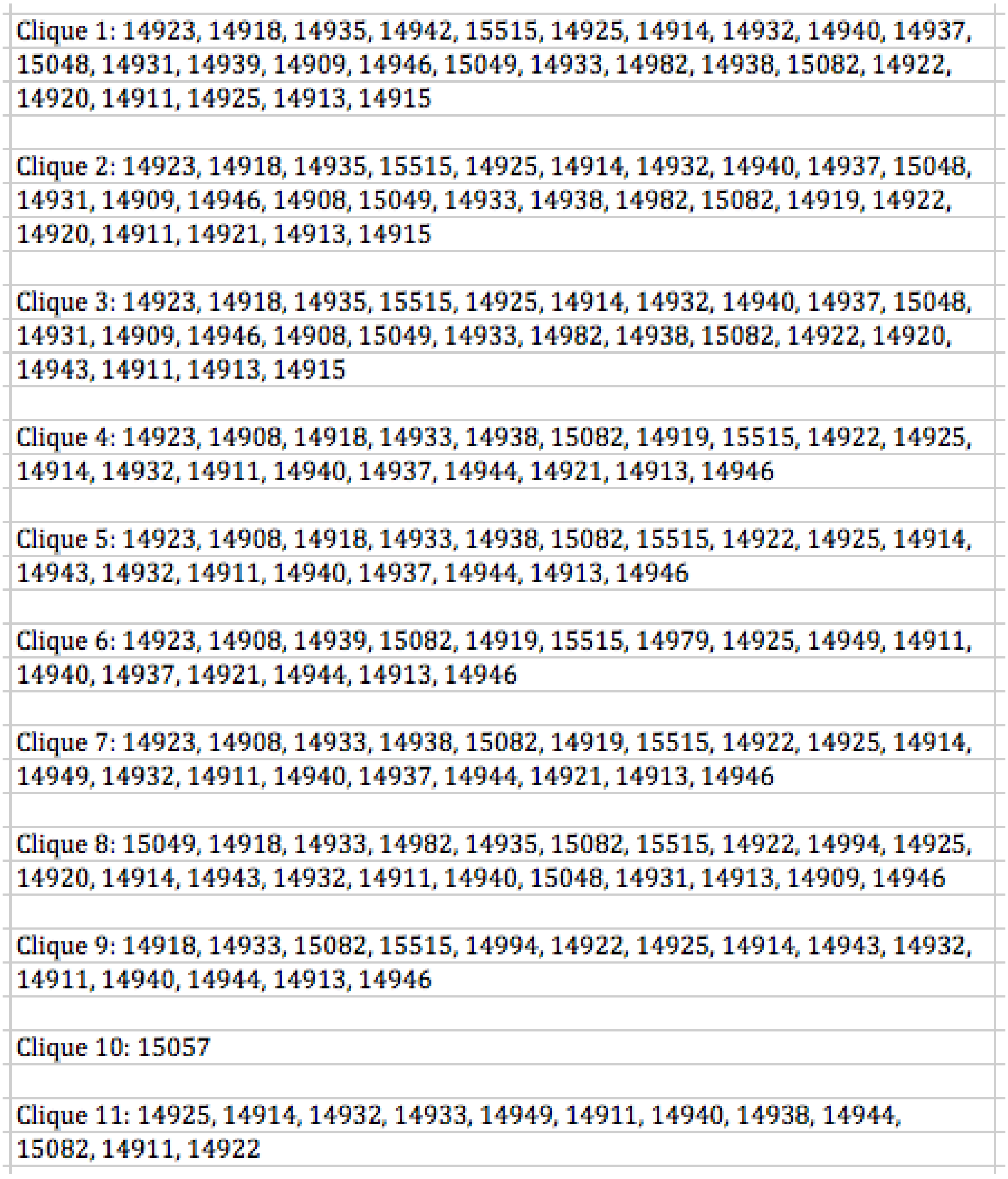
| | |
|---|---|
| Clique 1: 14923, 14914 | Clique 23: 14914, 14926, 14925 |
| Clique 2: 14914, 14908, 14918 | Clique 24: 14914, 14926, 14915 |
| Clique 3: 14908, 14918, 14949 | Clique 25: 14914, 14909, 14917 |
| Clique 4: 14914, 14918, 14909 | Clique 26: 14932 |
| Clique 5: 14914, 14939 | Clique 27: 14949, 14937 |
| Clique 6: 15057 | Clique 28: 14949, 14917 |
| Clique 7: 14937, 14935 | Clique 29: 14940 |
| Clique 8: 14982, 14935 | Clique 30: 14944 |
| Clique 9: 14935, 14942 | Clique 31: 14908, 14931 |
| Clique 10: 14949, 14942 | Clique 32: 14926, 14931 |
| Clique 11: 14911, 14942 | Clique 33: 14931, 14909 |
| Clique 12: 14931, 14942 | Clique 34: 14982, 14946 |
| Clique 13: 14942, 14915 | Clique 35: 15049 |
| Clique 14: 14942, 14946 | Clique 36: 14933 |
| Clique 15: 14949, 14922 | Clique 37: 14938 |
| Clique 16: 14994 | Clique 38: 15082 |
| Clique 17: 14936, 14946 | Clique 39: 14919 |
| Clique 18: 14914, 14908, 14911 | Clique 40: 14979 |
| Clique 19: 14914, 14937, 14946 | Clique 41: 14920 |
| Clique 20: 14914, 15048 | Clique 42: 14943, 14925 |
| Clique 21: 14914, 14913, 14909 | Clique 43: 14943, 14915 |
| Clique 22: 14914, 14913, 14915 | Clique 44: 14921 |

**Exhibit 8: Rhetorical Analysis - Subset of Docuscope Analysis**



| User ID | 14946 | 15048 | 14942 | 14979 | 14921 | 14926 | 14931 |
|---|---|---|---|---|---|---|---|
| [Personal_Register] | 6.21 | 7.92 | 7.1 | 8.33 | 6.39 | 4.55 | 5.49 |
| [Emotion] | 3.11 | 1.91 | 2.41 | 2.73 | 2.5 | 2.15 | 2.29 |
| [Assertive] | 5.25 | 4.51 | 4.89 | 4.67 | 5.51 | 3.9 | 3.31 |
| [Description] | 2.6 | 2.8 | 3.03 | 2.13 | 1.76 | 4.29 | 2.85 |
| [Public_Register] | 5.71 | 5.53 | 5.56 | 3.19 | 4.08 | 5.97 | 8.4 |
| [Academic_Register] | 8.38 | 9.62 | 8.12 | 7.13 | 6.02 | 8.99 | 9.71 |
| [Future] | 0.92 | 1.16 | 1.09 | 1.67 | 1.39 | 1.01 | 1.48 |
| [Past] | 3.07 | 1.84 | 1.9 | 2.55 | 1.34 | 2.04 | 1.53 |
| [Personal_Relations] | 2.88 | 1.91 | 2.34 | 1.85 | 2.09 | 2.17 | 2.31 |
| [Reasoning] | 2.32 | 2.53 | 2.32 | 2.27 | 2.64 | 2.2 | 1.32 |
| [Interactive] | 1 | 1.09 | 1.04 | 2.31 | 2.5 | 1.03 | 1.32 |
| [Elaboration] | 13.2 | 14.54 | 12.73 | 12.17 | 12.14 | 15.69 | 17.47 |
| [Reporting] | 7.85 | 9.15 | 7.94 | 8.24 | 9.73 | 9.1 | 8.96 |
| [Directives] | 0.02 | 0 | 0.05 | 0.05 | 0.14 | 0.1 | 0.08 |
| [Narrative] | 1.07 | 0.82 | 1.79 | 1.94 | 0.88 | 1.86 | 1.53 |
| [Character] | 9.72 | 6.35 | 9.01 | 5.92 | 5.51 | 8.84 | 8.07 |
| Total | 200.53 | 202.35 | 195.93 | 189.67 | 182.99 | 204.09 | 212.23 |

**Exhibit 9: Contextual Analysis - Cliques**



Clique 1: 14923, 14918, 14935, 14942, 15515, 14925, 14914, 14932, 14940, 14937, 15048, 14931, 14939, 14909, 14946, 15049, 14933, 14982, 14938, 15082, 14922, 14920, 14911, 14925, 14913, 14915

Clique 2: 14923, 14918, 14935, 15515, 14925, 14914, 14932, 14940, 14937, 15048, 14931, 14909, 14946, 14908, 15049, 14933, 14938, 14982, 15082, 14919, 14922, 14920, 14911, 14921, 14913, 14915

Clique 3: 14923, 14918, 14935, 15515, 14925, 14914, 14932, 14940, 14937, 15048, 14931, 14909, 14946, 14908, 15049, 14933, 14982, 14938, 15082, 14922, 14920, 14943, 14911, 14913, 14915

Clique 4: 14923, 14908, 14918, 14933, 14938, 15082, 14919, 15515, 14922, 14925, 14914, 14932, 14911, 14940, 14937, 14944, 14921, 14913, 14946

Clique 5: 14923, 14908, 14918, 14933, 14938, 15082, 15515, 14922, 14925, 14914, 14943, 14932, 14911, 14940, 14937, 14944, 14913, 14946

Clique 6: 14923, 14908, 14939, 15082, 14919, 15515, 14979, 14925, 14949, 14911, 14940, 14937, 14921, 14944, 14913, 14946

Clique 7: 14923, 14908, 14933, 14938, 15082, 14919, 15515, 14922, 14925, 14914, 14949, 14932, 14911, 14940, 14937, 14944, 14921, 14913, 14946

Clique 8: 15049, 14918, 14933, 14982, 14935, 15082, 15515, 14922, 14994, 14925, 14920, 14914, 14943, 14932, 14911, 14940, 15048, 14931, 14913, 14909, 14946

Clique 9: 14918, 14933, 15082, 15515, 14994, 14922, 14925, 14914, 14943, 14932, 14911, 14940, 14944, 14913, 14946

Clique 10: 15057

Clique 11: 14925, 14914, 14932, 14933, 14949, 14911, 14940, 14938, 14944, 15082, 14911, 14922