Analyzing Social Media Networks: Learning by Doing with NodeXL

Derek Hansen and Ben Shneiderman
University of Maryland

Marc Smith
Telligent Systems

Draft Version (7/07/09) with NodeXL Version 1.0.1.88

*Please do not distribute*

NodeXL – Network Overview, Discovery and Exploration for Excel
is an Excel 2007 Template for viewing and analyzing network graphs.

Available for free download at: [www.codeplex.com/nodexl](http://www.codeplex.com/nodexl)
# Table of Contents

## Acknowledgements

Introduction ........................................................................................................... 8

1) Basic: Getting started with NodeXL ................................................................. 9
   - Data entry: ........................................................................................................ 9
   - Showing the graph: .......................................................................................... 10
   - Highlighting an edge: ....................................................................................... 10
   - Importing an edge list: ..................................................................................... 11

2) Layout: Arranging Vertices in the Graph Pane ................................................ 12
   - Automatic Layout: ........................................................................................... 12
   - Directed Graph Type: ..................................................................................... 12
   - Updating the Graph Pane: ............................................................................... 12
   - Manual Layout: ................................................................................................ 13
   - Preserving manual layout: .............................................................................. 14
   - Zooming and Scale: ......................................................................................... 14

3) Visual Design: Making network displays meaningful ....................................... 15
   - Vertex Colors: ................................................................................................ 15
   - Adding Descriptive Data: ............................................................................... 16
   - Changing Vertex Size (and other properties): ................................................. 16
   - AutoFilling Columns: ...................................................................................... 17
   - Legend: ............................................................................................................. 19
   - Changing General Graph Appearance: ......................................................... 19

4) Labeling: adding text labels to vertices and links ........................................... 20
   - Adding Primary Labels: ................................................................................. 20
   - Adding Secondary Labels: ............................................................................... 21
Adding Tooltips: .................................................................................................................. 21

5) Graph Metrics: Calculating and visualizing metrics ........................................................................ 22

Computing Graph Metrics: ............................................................................................................. 22

Saving a NodeXL File: ..................................................................................................................... 22

Kite Network Example ...................................................................................................................... 23

Opening an existing NodeXL File: ..................................................................................................... 23

Overall Metrics: ................................................................................................................................ 24

Vertex Metrics: ................................................................................................................................. 24

Degree: ............................................................................................................................................... 25

Betweenness Centrality: ...................................................................................................................... 25

Closeness Centrality: .......................................................................................................................... 26

Eigenvector Centrality: ...................................................................................................................... 26

Clustering Coefficient: ....................................................................................................................... 26

6) Preparing Data: Merging Edges and Sorting to Label Data ................................................................ 27

SeriousEats Analysis .......................................................................................................................... 27

Merging Duplicate Edges: .................................................................................................................. 28

Sorting Data: ...................................................................................................................................... 29

Auto-Filling Data Columns: ................................................................................................................. 29

Formulas: ........................................................................................................................................... 31

7) Filtering: Reducing clutter to reveal important features .................................................................... 32

Dynamic Filters: ............................................................................................................................... 32

Filtering by Autofilling the Visibility Column: .................................................................................. 35

Subgraph Images: ............................................................................................................................... 37

Putting It All Together: ....................................................................................................................... 39

8) Clustering: Identifying and displaying vertex clusters .................................................................... 40

2007 Senate Voting Analysis ............................................................................................................ 40
Creating Clusters Manually: ................................................................. 42

Changing Advanced Layout Options: ................................................. 44

Creating Clusters Automatically: ......................................................... 44

Showing and Hiding Clusters: ............................................................. 45

9) Conclusion......................................................................................... 46

10) References and resources ............................................................... 47
Table of Figures

Figure 1: Starting with an empty Edges worksheet (left) and graph pane (right) ................................................................. 9

Figure 2: Seven friendships typed by hand, for example, Ann and Bob are friends ................................................................. 9

Figure 3: Your first graph shows the 8 friends and 7 friendships .......................................................................................... 10

Figure 4: Clicking on row 5 (Ann and Carol) highlights their friendship edge in the graph pane .............................................. 10

Figure 5 NodeXL Ribbon has sections for Data, Graph, Visual Properties, Analysis, Show/Hide, and Help .................. 11

Figure 6. Vertices for 8 friends in a circular layout .................................................................................................................... 12

Figure 7. Invitation graph with directed relationships, e.g., Ann inviting Bob to a party (shown by edges with arrows) ............................................................ 13

Figure 8. Network graph with 2 separate groups that emphasizes the importance of Carol who has given and received two invitations .................................................................................................................. 13

Figure 9. Color coding how shows women (pink) and men (blue) ............................................................................................ 15

Figure 10. The Vertices worksheet now includes user supplied columns for Age and number of Prior Parties ..... 16

Figure 11. Vertices can have properties such as Color, Shape, Size, and Opacity ................................................................. 16

Figure 12. Vertex sizes have been Autofilled based on the number of Prior Parties attended, revealing the wide disparity in social activity. The Legend at the bottom of the graph pane shows the Autofill for Size ............ 17

Figure 13. Autofill Columns dialog box used to set Vertex Size to the number of Prior Parties. To activate, be sure to click on the Autofill button at the bottom .................................................................................................................. 18

Figure 14. Vertex Size Options allow you to set the range for sizes. Setting the range to be from 1.5 to 7.0 ensures that all vertices are visible and avoids overlap of vertices .................................................................................................................. 18

Figure 15. Options dialog box shows current values for the visual properties of vertices and edges ............................. 19

Figure 16. Primary Label column is Autofilled with the Vertex name ......................................................................................... 20

Figure 17. Secondary labels are shown outside the vertices, so size coding can still be used ........................................... 21

Figure 18. Compute Metrics dialog box with all metrics selected ......................................................................................... 22

Figure 19. Kite Network shown with undirected edge list and manually created layout ..................................................... 23

Figure 20. Kite Network showing graph metrics mapped onto visual attributes ................................................................. 25

Figure 21. Serious Eats unmerged data with duplicate edges (e.g., rows 16, 18, and 20) that are displayed as a single edge connecting user cucumberpandan with Blog post GroceryNinja ................................................................. 27
Figure 22. Serious Eats merged data showing only one row connecting user cucumberpandan with Blog post GroceryNinja and a new Edge Weight column

Figure 23. Sorting the Vertex column in alphabetical order (Sort A to Z)

Figure 24. Using the automatic fill function after sorting to populate rows beginning with a “B_” as Blue Solid Diamonds

Figure 25. Serious Eats updated graph showing black disks as people, orange solid squares as Forum topics, and blue solid triangles as Blog topics

Figure 26. Dynamic Filters dialogue box that allows you to set minimum and maximum values to show

Figure 27. Dynamic Filters dialogue box after calculating the metric Degree and refreshing the filters

Figure 28. A dynamically filtered graph showing only edges with Edge Weight of 2 or higher, except the selected edge

Figure 29. Six images created by incrementally increasing the minimum Degree slider beginning with a minimum Degree of 1 (upper-left image) and ending with a minimum Degree of 6 (lower-right image)

Figure 30. Dynamic Filters set to a minimum of 6 Degree with Filter Opacity at 10%

Figure 31. Vertex Visibility Options dialog box

Figure 32. Autofilled Vertex Visibility Subgraph Images dialog box

Figure 33. Subgraph Images dialog box

Figure 34. Subgraph Images on the Vertices worksheet showing differences between forums such as Vietnamese and PerfectFood

Figure 35. Serious Eats visualization emphasizing most important people, forums, and blogs

Figure 36. Unfiltered 2007 Senate co-voting network showing all 48 senators connected to each other

Figure 37. Autofill Columns settings for 2007 Senate data with Edge Visibility set to “Greater Than 0.65”

Figure 38. 2007 Senate Data showing two clear clusters with a few boundary spanners in the middle

Figure 39. Cluster Vertices worksheet used to manually map Vertices to user created Clusters

Figure 40. Clusters worksheet

Figure 41. 2007 Senate co-voting network showing Republicans (Red), Democrats (Blue), and Independents (Yellow)

Figure 42. Layout Options dialogue box used to increase the repulsive force between vertices helping reduce overlap
Figure 43. Clusters worksheet after using Find Clusters to automatically detect clusters ................................................. 44

Figure 44. Automatically generated clusters showing 3 unique clusters ................................................................. 45

Acknowledgements

The authors would like to thank the many people who have made this document and the NodeXL project possible. First, the members of the NodeXL design and development team include Natasa Milic-Frayling, Eduarda Mendes Rodrigues, Janez Brank, and Annika Hupfeld from Microsoft Research in Cambridge, England, Tony Capone from Microsoft Research in Redmond, Washington, Eric Gleave from the University of Washington, Vladimir Barash from Cornell University, and Cody Dunne and Adam Perer from the University of Maryland. Support for NodeXL development has been generously provided by Tony Hey, Daron Green, and Dan Fay from the Microsoft Research External Research Programs group in Redmond, Washington.

We thank Serious Eats (http://www.seriouseats.com/) who has allowed us to use data collected from their fascinating online community, as well as Emily Mason who collected the Serious Eats dataset as part of her coursework. Special thanks to Chris Wilson of Slate Magazine for sharing the Senate 2007 voting data.

Our many users have provided remarkable feedback but Pierre de Vries merits a specific mention for pushing NodeXL beyond our expectations. Our research collaborators Dana Rotman and Elizabeth Bonsignore have made it possible to field test NodeXL and carefully document the results. The students of several classes who were assigned projects with NodeXL have been patient and forgiving as we refined the rough edges. We are grateful to these and many other people for their efforts to make NodeXL an easy and useful tool for understanding complex networks.
Analyzing Social Media Networks: Learning by Doing with NodeXL

Introduction

Social media tools such as email, discussion forums, blogs, micro-blogs, and wikis are used by billions of people worldwide. As they communicate through these media via desktop and web-based applications on fixed and mobile devices the result is the creation of multiple complex social network structures. The lively interaction and networks of relationships created through these technologies is of growing importance to individuals, organizations, and communities. Understanding how these social media networks grow, change, fail, or succeed is a growing concern to researchers and professionals. The field of social network analysis provides a set of concepts and metrics to systematically study these dynamic processes. The methods of information visualization have also become valuable in helping users to discover patterns, trends, clusters, and outliers, even in complex social networks.

The profusion of software tools for social network analysis and visualization demonstrates the strength of interest, but many of these tools are difficult to use, particularly for those who lack experience with programming languages. The open source software tool, NodeXL was designed especially to facilitate learning the concepts and methods of social network analysis with visualization as a key component (for more information see Smith, Shneiderman, et al. 2009).

The NodeXL Template for Microsoft Excel 2007 is a free and open source extension to the widely used spreadsheet application that provides a range of basic network analysis and visualization features. NodeXL uses a highly structured workbook template that includes multiple worksheets to store all the information needed to represent a network graph. Network relationships (i.e., graph edges) are represented as an “edge list”, which contains all pairs of vertices that are connected in the network. Other worksheets contain information about each vertex (i.e., node) and cluster. Visualization features allow users to display a range of network graph representations and map data attributes to visual properties including shape, color, size, transparency, and location.

NodeXL is designed to support students who are learning social network analysis and professionals interested in applying network analysis to business problems. It builds on the familiar spreadsheet paradigm to provide an easy to use tool for non-programmers. NodeXL integrates Excel’s native analysis functions, commonly used network metrics, and visualization to gain the benefit of all three approaches. It supports diverse visual network layouts, powerful filtering, clustering, and mapping of vertex and edge-level data onto highly customizable visual attributes and labels. The tool supports work with modest-sized networks of several thousand vertices, although some users have successfully dealt with tens of thousands of vertices.
1) **First steps: Getting started with NodeXL**

Get started by opening NodeXL at the Basic Layer which shows the usual Excel menu bar is across the top, a blank workbook on the left, and a graph pane on the right (Figure 1). NodeXL allows users to fill or paste in columns of edge list data in the Edges worksheet consisting of vertex pairs that are related to each other.

![Figure 1: Starting with an empty Edges worksheet (left) and graph pane (right)](image)

**Data entry:**

One way to begin using NodeXL is to type in your own edge list. For example, you might type the name of people who are friends in each row filling in the Vertex 1 and Vertex 2 columns (See Figure 2).

![Figure 2: Seven friendships typed by hand, for example, Ann and Bob are friends](image)
Showing the graph:
Click on the Show Graph button (directly above the graph pane) to show the network of friendships (Figure 3). The example assumes undirected relationships, that is, Ann is a friend of Bob, and Bob is a friend of Ann.

![Image of a network graph showing relationships between characters]

Figure 3: Your first graph shows the 8 friends and 7 friendships

Highlighting an edge:
Click one of the workbook rows to highlight the corresponding edge and the two vertices in the graph. For example, clicking on row 5 highlights the edge connecting Ann to Carol (see Figure 4). You can even click on multiple rows and all related edges and vertices will be highlighted.

![Image of a network graph with a highlighted edge]

Figure 4: Clicking on row 5 (Ann and Carol) highlights their friendship edge in the graph pane
Importing an edge list:

Another way to begin using NodeXL is to use the Import command to load an edge list from an existing file or data source. The Import Command is found on the NodeXL Ribbon (see Figure 5) along with other NodeXL specific commands. Someone may provide you with an edge list in the form of a Pajek file (another social network analysis program) or in a standard Excel workbook. Alternatively, you can cut and paste from another Excel spreadsheet to fill in the edge list. Additional import options (e.g., importing an email or Twitter network) are also available (see figure 5).

The NodeXL ribbon provides access to the core features, which you will be exploring later in this tutorial. Hovering over buttons displays additional information. Some features are accessible by right-clicking. You’ll be using the NodeXL controls to create meaningful layouts of vertices, controlling the visual properties of vertices and edges (e.g. color, size, opacity), and analysis methods.

Resizing and moving the Graph Pane:

As you work with the data you may want to resize the pane by moving the cursor to the left side of the pane until you see the ↔ symbol and then dragging it to the desired size. It is also possible to move graph pane to the left, above, or below the worksheet data by clicking on the title that reads “Document Actions” and dragging it around. You can even drag the graph pane outside the Excel window. When used on a computer with a large monitor or two or more monitors, the NodeXL graph pane can be moved to occupy one full screen while the spreadsheet is fully visible in another display.
2) Layout: Arranging Vertices in the Graph Pane

Automatic Layout:
NodeXL offers several automatic Layout Types that can be selected from the control in the graph pane. The default layout type for NodeXL is called Fruchterman-Reingold. A common alternate approach is to use the Circle Layout which spreads the vertices into a circle (Figure 6). In this case the two layouts are quite similar. Experimenting with different Layout Type (e.g. spiral, grid, Sugiyama) can reveal useful patterns, relationships, or unusual features in the data set being analyzed.

![Figure 6. Vertices for 8 friends in a circular layout](image)

Directed Graph Type:
In NodeXL, the default Graph Type is Undirected, which means the relationship between Vertex 1 and Vertex 2 is symmetric. In the example with friendships, if Ann is a friend of Bob, Bob is also a friend of Ann. This is shown in the graph pane as an edge connecting Ann and Bob. The NodeXL ribbon allows you to specify the Graph Type as Directed (see Figure 5), which means that relationships go only in one direction, for example the relationship might be that Ann has invited Bob to a party which does not require that Bob has also thrown a party and invited Ann. For directed graphs, Vertex 1 is the source and Vertex 2 is the destination. An arrow is shown leading into the destination vertex. Click on Refresh Graph (in the graph pane) to display the directed network (see graph pane in Figure 7).

Updating the Graph Pane:
Any time you change the underlying data or features that affect the layout of the network (e.g., directed versus undirected), you must click on the Refresh Graph button to update the graph. If you just want to change the layout you can select a new layout type and click on Lay Out Again to reduce processing time.
Manual Layout:

In our example graph composed of invitations you may want to move the vertices around to gain a better understand of the relationships. You can click and drag the vertices one at a time to create arrangements that emphasize structures or create a more orderly display (Figure 8). You can select multiple vertices by drawing a box around them or clicking on additional vertices while holding down the Control key. If multiple vertices are selected they will all move together when dragged.
**Preserving manual layout:**
After working to get a layout that shows important relationships, you may want to preserve that layout. In the layout selection menu chose “None”, which keeps your manual layout, even after selecting Refresh Graph. Another more permanent method for fixing vertex placement is described in the Advanced Feature box below.

**Zooming and Scale:**
To get a closer look at a subsection of a graph you can use the Zoom slider (or a mouse scrollbar in the graph pane). Once you are zoomed in you can pan across the graph by holding down the Spacebar, clicking the mouse button, and dragging the cursor in the direction you want to pan. You can also use the Scale slider to change the size of the vertices and edges.

---

**Advanced Feature:**

**Fixing Vertex Placement:** You can fix the placement of the vertices so they do not change when you click on Refresh Graph, even if an automatic layout other than None is chosen. First, click on the Workbook Columns button on the NodeXL Ribbon and check Layout from the list. This will display the Layout related columns in the Edges and Vertices worksheets that are hidden by default. Next find the “Locked?” column on the Vertices tab and choose “Yes (1)” (or just “1”) for each of the vertices. You can also use the two columns labeled X and Y to fine-tune vertex placement if desired. For example, you could set the Y values of certain vertices to the same number to assure that they line up perfectly.
3) Visual Design: Making network displays meaningful

Drawing a meaningful graph can reveal patterns, relationships, and interesting features that may be hard to spot in a tabular edge list. NodeXL is designed to enable you to create a rich variety of possible drawings for a graph.

**Vertex Colors:**

You may want to change the colors of vertices. For example, in the friendship graph, you might want to color vertices that represent men with blue and the women with pink. Look at the worksheets on the lower left and click on the Vertices worksheet, which will bring up the list of 8 vertices (also called nodes) in our party invitation data set. The contents of the Vertices tab are generated automatically from the Edges data. Choose the color you want for each person from the drop-down menu available from each cell of the Color column. Alternatively, after selecting a person, click on the Color button in the NodeXL Ribbon’s Visual Properties section and select the color you want from the color palette. You can even click on multiple vertices using the Ctrl and/or Shift keys and set all their visual properties together. Once you’ve populated the Color column, click on the Refresh Graph to redisplay the Graph Pane (Figure 9).

![Color coding how shows women (pink) and men (blue)](image-url)
Adding Descriptive Data:
If you have additional information about the people in the data set, you can add your own columns of data by typing (or pasting it in). To record the age of each person, scroll the Vertices worksheet to the right until you see the column header “Add your own Columns Here.” Place the cursor on this header to get further instructions. If you select the next free column, you can type an attribute name (e.g., Age) and then enter values for each person. Add two new columns, one for Age and one for the number of Prior Parties the individual has attended since the beginning of the year as shown in Figure 10.

![Figure 10. The Vertices worksheet now includes user supplied columns for Age and number of Prior Parties](image)

Changing Vertex Size (and other properties):
Another visual property that can be used to encode attribute values is vertex size, which is controlled by the Size column in the Vertices worksheet. Put your cursor over the Size column header to show the type of data that must be entered – in this case numbers 1-10. Use this same approach to see what type of data to enter into any of the different fields such as Shape, Color, and Opacity (Figure 11).

![Figure 11. Vertices can have properties such as Color, Shape, Size, and Opacity](image)
There are three ways to enter numbers into the Size column (or other visual attributes such as Opacity or Color): (1) You can manually type them in, (2) you can enter a formula that calculates a number for the Size based on some other data (e.g., the Prior Party field you entered earlier), or (3) you can use the AutoFill feature to let NodeXL fill in the column based on some other data (e.g., the Prior Party column). Figure 12 shows the result of using the NodeXL Autofill feature to automatically fill in the Size numbers based on the Prior Parties data you entered earlier.

Figure 12. Vertex sizes have been Autofilled based on the number of Prior Parties attended, revealing the wide disparity in social activity. The Legend at the bottom of the graph pane shows the Autofill for Size

**AutoFilling Columns:**

To recreate Figure 12, first click on the AutoFill Columns button in the NodeXL ribbon. The resulting Dialog box (Figure 13) offers a set of drop-down boxes to allow you to select data you have entered in as additional fields. Click on the symbol next to Vertex Size to see all of the data columns you have entered in and choose Prior Parties (instead of Age). You can do the same for many other visual attributes of the Vertices as well as the Edges. Those associated with vertices populate columns in the Vertices worksheet, while those associated with edges populate columns on the Edges worksheet. The column data will show up when you click on Refresh Graph.
Each attribute has an associated Options page that allows you to fine-tune some of the attributes. In our example, we want to assure that the vertices are large enough to view well, so we can click on the  button in the Options column for the Vertex Size row (Figure 14).

Figure 13. Autofill Columns dialog box used to set Vertex Size to the number of Prior Parties. To activate, be sure to click on the Autofill button at the bottom.

Figure 14. Vertex Size Options allow you to set the range for sizes. Setting the range to be from 1.5 to 7.0 ensures that all vertices are visible and avoids overlap of vertices.
Legend:
Each time you use Autofill, NodeXL adds to the legend which is shown at the bottom of the Graph Pane. This legend helps you and your viewers to understand the visual properties of the graph. In our example, the size property was set by Autofill, so the legend shows that maximum size for Prior Parties is 7. Because color was manually entered, it does not show up in the legend.

Changing General Graph Appearance:
Another way of setting visual features is to go to the Graph Pane and click on the Options button (or right click in the Graph Pane and select Options) to bring up the Options Dialog Box (Figure 15). It offers controls for setting the default visual features for Vertices, Selected vertices, Edges, Selected edges, Fonts, Margins, etc. Default visual properties (e.g., Color, Shape, Opacity) will be superseded by numbers in the corresponding columns on the Vertices or Edges worksheets if they are populated.

Figure 15. Options dialog box shows current values for the visual properties of vertices and edges
4) Labeling: adding text labels to vertices and links

Since textual labels are helpful in understanding graphs, NodeXL offers three ways to display them, all of which can be used simultaneously:

**Primary labels**: Text such as the vertex name appears inside the vertex in a rectangular box. Color and Opacity can still be used, but Shape and Size cannot.

**Secondary labels**: Text appears outside of the label, enabling you to use all visual properties including Shape and Size, but adding to the potential for screen clutter.

**Tooltip**: Text appears as a pop-up only when your cursor hovers over the vertex. This keeps the graph pane uncluttered, but only allows you to see text associated with one vertex at a time.

To set up text labels, go to the NodeXL ribbon, and in the Show/Hide group, select the Workbook Columns button, then check the Labels entry. This will make the necessary columns visible in the Vertices worksheet.

**Adding Primary Labels**:

You can invoke the AutoFill Columns feature to fill the Primary Label column with the names from the Vertex or another column. Then, when you click on Refresh Graph, the vertices become filled with the labels (Figure 16). The color coding remains but the size coding is no longer used. In this case the Pink color made the text too light to read easily, so the color Pink was changed to Deep Pink.
Adding Secondary Labels:
You can show labels outside the vertex by using Secondary Labels, thereby allowing characteristics such as Size and Shape to be used for the vertices. To re-create Figure 17, use the AutoFill feature to fill the Secondary Label column with the Vertex column. Clear the Primary Label column by highlighting all data cells and using the Delete key or right-clicking and selecting Clear Contents. In Figure 18, the Options dialog box (Figure 15) was used to set the default Font Size to 12 point. You can also make the Edges semi-transparent so labels that overlap with them will be more readable. To do so, set the Edges Opacity to 40 within the Options dialog box (Figure 15).

![Figure 17. Secondary labels are shown outside the vertices, so size coding can still be used](image)

Adding Tooltips:
You can also add data that only shows up when you mouse over a vertex. This is called a Tooltip. In Figure 17, the AutoFill has been used to associate the Tooltip column with the Age column. When you mouse over Helen you will see her age (22 in this case).
5) **Graph Metrics: Calculating and visualizing metrics**

When trying to understand networks, analysts often want to identify important vertices, locate subgroups, or get a sense of how interconnected a network is compared to other networks. While visualization itself can help do this, it is often helpful to use graph metrics that provide quantitative measures that characterize various aspects of a graph. NodeXL can calculate several graph metrics for you. Once calculated, you can use the graph metrics to change the visual display of your network graphs in powerful ways.

**Computing Graph Metrics:**

To calculate graph metrics first click on the Graph Metric button on the Analysis section of the NodeXL Ribbon. This will open up the dialogue box in Figure 18 that shows you the available graph metrics. Select the ones you want to calculate by checking in the boxes next to them. Clicking on the Details link next to a metric provides a brief explanation of that metric. Click on the Select All button and then choose Compute Metrics. Some of the graph metrics can take a while to calculate when working with large networks, so a status bar is used to show progress. Once completed, NodeXL displays each vertex-specific metric in a new set of Graph Metrics columns in the Vertices worksheet. NodeXL also populates the Overall Metrics worksheet showing summary information for the entire network if Overall Metrics were calculated.

**Figure 18. Compute Metrics dialog box with all metrics selected**

**Saving a NodeXL File:**

You are now done with the party example used up to this point. To save the NodeXL file, save it as you would any other Excel file making sure to select the standard Excel Workbook (with a .xlsx extension). Do not save it as an Excel 97-2003 Workbook, a Macro-Enabled Workbook, or a Binary Workbook.
Kite Network Example

To better understand the meaning of the various graph metrics, you will now begin using a network called the Kite Network, created by David Krackhardt (see http://www.orgnet.com/sna.html). You can download the Kite_Network.xlsx file from: http://casci.umd.edu/NodeXL_Teaching or you can manually reproduce the undirected edge list and graph shown in Figure 19 in a new NodeXL template. The download version has fixed the position of the vertices to match those found in Figure 19.

**Opening an existing NodeXL File:**

You can open a NodeXL file just as you would any other Excel file. If NodeXL is installed on the machine, Excel will recognize any file created using NodeXL even though it has the standard .xlsx extension. Opening the file will automatically launch NodeXL. Once you have opened the file, select Show Graph and then calculate all of the Graph Metrics.

![Figure 19. Kite Network shown with undirected edge list and manually created layout](image-url)
**Overall Metrics:**

Go to the Overall Metrics worksheet, which summarizes some of the key properties of the entire network including the following:

- Graph Type: undirected or directed
- Unique Edges: number of unique edges entered into the Edges worksheet
- Edges with Duplicates: number of repeated vertex pairs on the Edges worksheet. Duplicate vertex pairs may occur, as for example in a discussion forum network when Person A replies to Person B on multiple occasions. Duplicate vertex pairs can cause some metrics such as Degree to be inaccurate. They can be combined into a single weighted edge by choosing the Merge Duplicate Edges as described later in this tutorial.
- Total Edges: number of total edges, i.e., rows on the Edges worksheet.
- Self-Loops: number of edges that connect a vertex with itself. A self-loop occurs when the edge list includes the same exact name in the Vertex 1 and Vertex 2 columns on the Edges tab (i.e., a person is connected to themselves). This may happen when, for example, in an email list edge list a person replies to their own email. Self-loops are represented visually in the graph pane by a circular edge that comes out of a vertex and returns to that same vertex.
- Vertices: number of total vertices, i.e., rows on the Vertices worksheet.
- Graph Density: number between 0 and 1 indicating how inter-connected the vertices are in the network. For an undirected graph where all vertices are connected to all others through at least one edge, the Graph Density is calculated by dividing the number of Total Edges by the maximum number of possible edges. For the Kite network there are 18 edges and 45 possible edges, resulting in a Graph Density of 0.4. A more dense graph (e.g., 0.6) would include more Total Edges for a comparable number of vertices.
- NodeXL Version: indicates the version of NodeXL being used when Metrics were calculated.

**Vertex Metrics:**

To see the vertex-specific metrics such as centrality measures and clustering coefficients go to the Vertices worksheet. You will see the new Graph Metrics columns, which can be hidden later if desired by un-checking Graph Metrics from the Workbook Columns button on the NodeXL Ribbon. Each value relates directly to one of the vertices. For example, row 2 shows the various graph metrics that are specific to Andre (Figure 20).
Vertex metrics can be mapped onto visual attributes as shown in Figure 20, which you can recreate by using the Autofill Columns feature. The graph legend shows that Degree is mapped to Size and Betweenness Centrality is mapped to Opacity. In addition, Closeness Centrality is mapped to the Tooltip. Below is a description of each metric and how it relates to the Kite network.

**Degree:**

The Degree of a vertex (sometimes called Degree Centrality) is a count of the number of edges that are connected to it. Diane has a Degree of 6 because she is directly connected to 6 other individuals. In comparison, Jane has a Degree of only 1 because she is connected to only 1 other person. If the edges represented strong friendship ties of individuals in a class, we might say that Diane is the most popular person in the class and Jane is the least popular. The legend in Figure 20 shows the range of the Degree (1 to 6) mapped onto size. The size of the vertices has been set using the Autofill Size Options to a range of 2 to 7 so the vertices are clearly visible, but not too large. If we were using a directed graph (such as the Party Network), the single Degree metric would be split into two metrics: (1) In-Degree, which measures the number of edges that point toward the vertex of interest (i.e., number of people that have invited the person to the party), and (2) Out-Degree, which measures the number of edges that the vertex of interest points toward (i.e., number of people the person has invited to the party).

**Betweenness Centrality:**

While popularity is important, it is not everything. Consider Heather in the Kite network. She is only directly related to 3 other people (i.e., she has a degree of 3). Despite her relatively low Degree, her...
position as a “bridge” between Ike (and indirectly Jane) to the rest of the group may be of utmost importance. If, for example, information were passed from one person to another, Heather would be vital for assuring that Ike and Jane could communicate with the rest of the group. In fact, if she was removed from the network, Ike and Jane would be disconnected from the other class members. Thus, Heather has high Betweenness Centrality. In contrast, Ed has a Betweenness Centrality of 0. Notice that if he were removed from the graph everyone would still be connected to everyone else and their shortest communication paths would not even be altered. More generally, vertices that are included in many of the shortest paths between other vertices have a higher Betweenness Centrality than those that are not included. In Figure 20 the legend shows that the AutoFill feature has set the Opacity of each vertex to the Betweenness Centrality metric, which ranges from 0 (Ed and Carol who show up lighter) to 1 (Heather who shows up darkest). To make sure each vertex is visible, the minimum Opacity was set to 40 and maximum was kept at 100.

**Closeness Centrality:**

Another characteristic you may care about is how close each person is to the other people in the network. If information flowed through edges in the network, some people would be able to contact all the other people in only a few steps, while others may require many steps. Closeness Centrality is a measure of the average shortest distance from each vertex to each other vertex. Unlike other centrality metrics, a lower Closeness Centrality score indicates a more central (i.e., important) position in the network. In the Kite Network, Fernando and Garth have the lowest Closeness Centrality measure, suggesting that they may be in a good position to spread information through the network efficiently. In Figure 20 the AutoFill feature was used to set the Tooltip to the Closeness Centrality metric (notice the number 2 that shows up when hovering the mouse over Ed).

**Eigenvector Centrality:**

In many cases, a connection to a popular individual is more important than a connection to a loner. The Eigenvector Centrality metric takes into consideration not only how many connections a vertex has (i.e., its Degree), but also the Degree of the vertices that it is connecting to. Both Heather and Ed have a Degree of 3. However, Ed is directly connected to Diane, the most popular person in the class, whereas Heather is connected to Ike who is among the least popular. This explains why the Eigenvector Centrality metric for Heather is lower than it is for Ed.

**Clustering Coefficient:**

In some cases, a person’s friends may be friends with each other, creating a clique. For example, Ed’s three friends Beverly, Diane, and Garth are all directly connected to one another (i.e., they create a complete graph). In other cases, a person’s friends may not be friends with one another. For example, Ike’s two friends Heather and Jane are not friends with each other. The Clustering Coefficient measures how connected a vertex’s neighbors are to one another. More specifically, it is the number of edges connecting a vertex’s neighbors divided by the total number of possible edges between the vertex’s neighbors. For example, Heather’s 3 neighbors are Fernando, Garth, and Ike. Only one connection exists between any of them (the connection between Fernando and Garth). There are 3 possible connections (Fernando-Garth; Fernando-Ike; Garth-Ike). Thus, the Clustering Coefficient for Heather is 1/3.
6) Preparing Data: Merging Edges and Sorting to Label Data

The examples so far have used small, simple networks with only a handful of vertices. Most social media networks are much larger, often creating cluttered graphs that are hard to interpret. NodeXL includes powerful strategies for making sense of these larger networks and discovering important features of the data, but to take advantage of these it is often necessary to prepare the initial data.

SeriousEats Analysis

This section analyzes a network generated from discussion forum posts and blog comments made to the SeriousEats online community by food enthusiasts (http://www.seriouseats.com). Data were manually collected from publicly accessible content taken from the SeriousEats website on March 7-8, 2009 by Emily Mason. You will need to download the data from the file titled “Serious_Eats.xlsx” found at: http://casci.umd.edu/NodeXL_Teaching. The file includes only an edge list. Vertex 1 includes the usernames of community members who have contributed to the site. Vertex 2 includes abbreviated names of discussion forums or blog posts that the community members posted to. Blog posts begin with a “B_” and discussion forum posts begin with a “F_”. For example, the first row shows that user gastronomeg posted to the Blog entry with the abbreviated title Misosoup (Figure 21). This type of dataset with Vertex 1 representing people and Vertex 2 representing some event (i.e., posting in a forum or blog) is an example of “affiliation data.” More generally, a network with two different entities represented in Vertex 1 and Vertex 2 columns is called a “bi-modal” network or “two mode” network.

Figure 21. Serious Eats unmerged data with duplicate edges (e.g., rows 16, 18, and 20) that are displayed as a single edge connecting user cucumberpandan with Blog post GroceryNinja
**Merging Duplicate Edges:**

You may notice that some rows are duplicates (rows 16, 18, and 20 in Figure 21). This is not an error since some community members posted multiple times to the same forum or blog. For example, user cucumberpandan posted to the Blog GroceryNinja on 3 separate occasions. However, as shown by the red highlighting in the graph pane of Figure 21, only 1 edge is shown for each of the duplicate rows. NodeXL allows you to remove the duplicate edges, while retaining information about how many times an edge was duplicated. Click on the Merge Duplicate Edges button in the Prepare Data dropdown menu on the NodeXL Ribbon as shown in Figure 21 and then Refresh the graph.

You will now see a new column called Edge Weight that indicates the number of edges that were rolled up (i.e., merged). As shown in Figure 22, there is now only one row connecting cucumberpandan with B_GroceryNinja showing an Edge Weight of 3, since 3 original rows were merged into 1. In total, the original 417 unmerged edges are now condensed into 362 merged edges.

![Figure 22. Serious Eats merged data showing only one row connecting user cucumberpandan with Blog post GroceryNinja and a new Edge Weight column](image)

The graph shown in Figure 22 is not easy to interpret, largely because it includes so many vertices and edges. It also doesn’t make clear the fact that some vertices represent different things than other vertices. To resolve this issue, you can set unique shapes and colors to each of the different types of vertices. This can be done manually with the aid of sorting.
**Sorting Data:**
NodeXL can take advantage of Excel’s native support for sorting columns. This can be used to help annotate data efficiently and identify important vertices. Go to the Vertices worksheet and click on the drop-down menu triangle in the Vertex label cell of the first column. Select “Sort A to Z” from the menu (Figure 23). This will sort all of the Vertices alphabetically, which groups all of the blog posts (beginning with “B_”) and discussion forum posts (beginning with “F_”) next to each other making it easy to set unique color and shape attributes for each group.

![Figure 23. Sorting the Vertex column in alphabetical order (Sort A to Z)](image)

**Auto-Filling Data Columns:**
Set the Color to Black and Shape to Disk (2) for the people (rows 1-15 as shown in Figure 24). To efficiently fill in the column cells you can enter the desired color and shape in the first row, highlight both cells, and move the cursor to the bottom-right corner of the Shape cell until it becomes a symbol. Drag this symbol down and the cells will all fill in with the same content. When you get to the first row beginning with a “B_” (B_FoodGlossies), change the Color to Blue and the Shape to Solid Diamond (7). Use the same fill shortcut to populate all rows beginning with a “B_” as shown in Figure 24. Populate the
remaining usernames as Black Disks and the forum posts beginning with “F_” to Orange Solid Squares. Refresh the graph to see the results (Figure 25).

![Graph showing network analysis with NodeXL](image1)

**Figure 24.** Using the automatic fill function after sorting to populate rows beginning with a “B_” as Blue Solid Diamonds

![Graph showing updated Serious Eats graph](image2)

**Figure 25.** Serious Eats updated graph showing black disks as people, orange solid squares as Forum topics, and blue solid triangles as Blog topics
**Formulas:**
You can use Excel’s built-in functions to calculate values in any of the cells. For example, you can enter formulas in the Color and Shape columns to automatically do what you just did manually. The formulas would look for unique text strings in the Vertex column (e.g., “B_” and “F_”) and use logic such as “if” statements to set them appropriately. Functions available from Excel’s Formula ribbon in the Textual, Logical, and Lookup & Reference categories are particularly helpful when using NodeXL. This tutorial does not require you to know functions, but they are a powerful tool for those who know them or are willing to experiment with them.

Because AutoFill was not used to populate the Color and Shape columns, the legend does not indicate the meaning of the colors. You may want to create your own key to describe the mapping.
7) Filtering: Reducing clutter to reveal important features

When working with large, cluttered graphs it is often useful to filter out vertices or edges or to focus only on sections of the larger graph (i.e., sub-graphs). NodeXL offers a variety of ways to filter out edges and vertices that will be presented in this section using the Serious Eats dataset.

Dynamic Filters:
Filtering out certain edges or vertices so they don’t show up on the graph is a good way to reduce clutter. One way to use the Dynamic Filters feature accessible via buttons in the NodeXL Ribbon’s Analysis section or just above the graph pane (you may have to click on the downward pointing arrow on the upper-right hand side of the graph to access the Dynamic Filters button). This will open a new dialogue box (Figure 26). The box offers a number of double box range sliders to help you filter. The number on the left-hand side is the minimum value found in the workbook, while the number on the right-hand side is the maximum value. The top set of sliders filter out Edges, leaving in the Vertices. The second set of sliders filter out the vertices and all edges that point to those vertices.

[Image: Dynamic Filters dialogue box]

Figure 26. Dynamic Filters dialogue box that allows you to set minimum and maximum values to show

New filters appear when additional metrics are calculated or new columns are added with data. Calculate the metric “Degree” as described earlier in the tutorial. Then click on the Read Workbook button in the Dynamic Filters dialogue box (Figure 26). You will now see a new slider titled “Degree” in the Vertex Filter’s area as shown in Figure 27. Try filtering sliding the Edge Weight slider on the left-hand side to the right so that the number changes from 1 to 2. The graph should be dynamically updated so that only edges that have an edge weight of 2 or higher will be displayed. The resulting graph (Figure 28) only shows ties where a person has posted to a forum topic (or blog post) 2 or more times.
When items are filtered, they are still read into the graph and will show up if you click on the corresponding vertex or edge in the data portion of the spreadsheet. This is demonstrated in Figure 28 where the edge connecting gastronomeg and the blog post titled MisoSoup are shown in red even though their Edge Weight is less than 2.

![Dynamic Filters dialogue box](image)

**Figure 27. Dynamic Filters dialogue box after calculating the metric Degree and refreshing the filters**

![Dynamically filtered graph](image)

**Figure 28. A dynamically filtered graph showing only edges with Edge Weight of 2 or higher, except the selected edge**
Click on the Reset All button in the dynamic filters dialogue box (Figure 27) to show all of the edges and vertices. Next, click on the upward pointing arrow on the left-hand side of the Degree slider. This will incrementally remove vertices with a Degree smaller than the number in the left-hand box. Figure 29 shows a series of graphs starting with all vertices and continuing to remove vertices with Degree of 1, then 2, then 3, and so forth. The graph images were copied to the clipboard by right-clicking on the graph pane and selecting Copy Image to Clipboard from the menu. Images can also be exported from the same menu in a variety of formats.

![Figure 29. Six images created by incrementally increasing the minimum Degree slider beginning with a minimum Degree of 1 (upper-left image) and ending with a minimum Degree of 6 (lower-right image)](image)

These graphs make clear that most people (black disks) are connected to only 1 or 2 forum or blog posts during the time frame of data collection, and most forum posts (orange squares) are connected to at least 6 people.

You can set the Filter Opacity to show the filtered out edges while still making them less prominent. Enter 10 into the Filter Opacity box on the Dynamic Filters dialogue box (Figure 27) to recreate Figure 30. Even when the Filter Opacity is 0, the vertices and edges are retained in the graph, they are just hidden. For example, if you try and layout the graph again after reducing the vertices, the layout will not change significantly because it is laying out the graph using all of the edges and vertices.

When dynamic filters are used, the legend at the bottom of the graph pane is updated to reflect the settings as shown in Figure 30.
Filtering by Autofilling the Visibility Column:

Another method of filtering is to use the Autofill Columns feature already introduced to automatically set the Visibility Column. Before trying this, choose Reset All on the Dynamic Filters dialogue box (Figure 27). Next, open the Autofill Columns dialogue box, select Degree in the drop-down menu for Vertex Visibility, and choose the arrow to the right that opens the Vertex Visibility Options dialogue box shown in Figure 31.
A number of options are available by clicking on the drop-down menu as has been done in Figure 31. Select the “Greater than or equal to” option, replace the value 0 with 6, and click OK. Unlike dynamic filters, when you use this method, only the unfiltered vertices (or edges) are read into the graph. As a result, you can choose different layouts and they will reposition the visible vertices and edges as if they were the only ones. This is shown in Figure 32, where the Sugiyama layout was used. Additionally, if you select a vertex or edge, only those that already appear in the graph will be turned red, since there are no hidden vertices or edges. For example, in Figure 32 the vertex representing the Forum CheffTell is selected, which has a Degree of 14 indicating that 14 unique people have contributed to that forum. However, only one person is connected to the forum in the current graph. The other 13 people were not read into the graph since they had a Degree of less than 6.

Open the Autofill Columns dialogue box and set the Vertex Visibility Options dialogue box (Figure 31) to read “Greater than or equal to” 0 instead of 6. Click OK, and then Autofill the columns. Select the layout type Fruchterman-Reingold and choose the Lay Out Again button at the top of the graph pane. You should have something that looks like the earlier graph shown in Figure 25.

Figure 32. Autofilled Vertex Visibility Subgraph Images dialog box
**Subgraph Images:**

Another useful way to understand complex networks is to view individual sections of the larger graph. NodeXL allows you to create sub-graph images for each vertex. Go to the Analysis section on the NodeXL ribbon and click on the Subgraph Images button. The Subgraph Images dialog box (Figure 33) will appear. The first option allows you to choose the levels of adjacent vertices to include in each subgraph. For example, the default of 1.5 will show edges connecting the source vertex with its direct neighbors, as well as any edges that connect the neighbors to one another. Choosing 2.0 will show all of those edges, plus edges connecting the source vertex’s neighbors with all of their neighbors. If the data were from a social networking site such as Facebook, a 2.0 setting would show your friends, which of your friends know one another, and all of your friends’ friends (FOAF). For now, replicate Figure 33 by choosing 2.0, checking the boxes that specify to select the vertex and vertex’s incident edges, and clicking Create. This will generate a new column called Subrgraphs as shown in Figure 34.

![Subgraph Images dialog box](image)

*Figure 33. Subgraph Images dialog box*

Other options on the Subgraph Images dialogue box (Figure 33) allow you to change the size of the images, export them as new image files, and only create subgraph images for vertices that are highlighted on the Vertices worksheet.
These subgraphs highlight important differences between vertices. To illustrate this point, sort on the Vertex column on the Vertices worksheet (from A to Z) and scroll down to those vertices beginning with “F_”. Compare the subgraphs for F_Vietnamese and F_PerfectFood (Figure 34). The F_Vietnamese image makes clear that F_Vietnamese discussion occurs between people who don’t frequent other discussion forums or posts. In contrast, the F_PerfectFood forum includes many people who have posted to other forums and blog posts. Similar comparisons can be made for blogs (beginning with “B_”) and people.
Putting It All Together:

Combining the various approaches in this and prior sections you can recreate Figure 35, which presents a much more readable graph than our original graph shown in Figure 21. Autofill was used to set Visibility to “Greater than or equal to 2”, vertex Size (1.5 to 4) was mapped to Degree, Edge Width (1 to 3) was set to Edge Weight, and Edge Opacity (50 to 100) was set to Edge Weight. Dynamic Filters were set to a Filter Opacity of 5 and set to filter out vertices with a Degree of less than 4. Vertices were manually adjusted to more easily make boundary spanners (i.e., those who post to both blogs and discussion forums) more obvious. A secondary label was manually entered for vertices with the highest Degree. Figure 35 makes clear that most people that few people post to multiple blogs, many post to multiple forums or a blog and a forum, and there are a few forums and one blog that solicit significant participation compared to others.

![Figure 35. Serious Eats visualization emphasizing most important people, forums, and blogs](image-url)
8) Clustering: Identifying and displaying vertex clusters

It is often helpful to identify vertices that are clustered together into subgroups of interest. Sometimes you will know which people should be classified into different clusters (e.g., Republicans versus Democrats), while other times you may want to identify clusters that you don’t know to look for ahead of time (e.g., friendship cliques within a large social network). NodeXL allows you to create your own clusters manually. It can also help automatically identify clusters of interest for you. Once identified, the color and shape of the vertices can be customized to visually display the clusters. To demonstrate how clusters work, you will analyze the voting patterns of U.S. Senators in the year 2007. You will also put together some of the concepts you’ve learned earlier. Special thanks to Chris Wilson of Slate Magazine for providing the dataset that can be downloaded from: http://casci.umd.edu/NodeXL_Teaching titled Senate_Raw.xlsx.

2007 Senate Voting Analysis

The Vertices worksheet includes data about each Senator including their party affiliation, the State they represent, and the total number of votes they cast in 2007. The Edges worksheet includes an undirected edge list connecting each senator to each other senator. The added columns shown in Figure 36 indicate the total number of votes that were the same (i.e., both voted Yea or both voted Nay) (Voted Same column), the total number of votes cast by the person in Vertex1 (Vertex1_Total) and Vertex2 (Vertex2_Total), and the percent agreement (Percent_Agreement). The lowest of the two Total Votes (columns K and L in Figure 36) is used as the denominator when calculating Percent_Agreement to help deal well with data from frequent absentees (e.g., those campaigning).

![Figure 36. Unfiltered 2007 Senate co-voting network showing all 48 senators connected to each other](image-url)

Showing the graph results in a large black mass of connections (Figure 36). This is because every senator is connected to every other senator at least once. To make sense of the data you will need to filter some of the edges and change some of the visual components.
Start by changing the color of all of the edges to be Light Gray by finding the Color column on the Edges worksheet, typing in “Light Gray” and copying it down to the last edge. Next, open the AutoFill Columns window and select the fields that match those in Figure 37. Set the Option for Edge Visibility to “Greater Than 0.65” (the average agreement percentage between all pairs of senators). The result is that pairs who voted the same less than 65% of the time will not be connected in the graph. As discussed in the Filtering section, they will not be read into the graph either (i.e., they will be “skipped,” not “hidden”). Because they are not read into the graph, the calculation of graph metrics and clusters treats them as if they don’t exist, which is desirable in this case. Autofill the columns to reveal an image like the one shown in Figure 38.

![Autofill Columns](image)

**Figure 37. Autofill Columns settings for 2007 Senate data with Edge Visibility set to “Greater Than 0.65”**
Creating Clusters Manually:

To manually create a cluster, go to the Cluster Vertices worksheet (Figure 39). Copy and paste the Vertex column from the Vertices worksheet into column B (Vertex). Then copy and paste the Party column from the Vertices worksheet to Column A (Clusters). Each of the Vertices is now assigned to a cluster based on their Party affiliation.

Figure 38. 2007 Senate Data showing two clear clusters with a few boundary spanners in the middle

Figure 39. Cluster Vertices worksheet used to manually map Vertices to user created Clusters
Go to the Clusters worksheet and type in the information shown in Figure 40. This determines the Color and Shape of each vertex assigned to a cluster. Make sure the Clusters listed in Column A include all of the unique values in the Cluster Vertices worksheet (Figure 39). When the check mark next to Clusters in the Show/Hide section of the NodeXL ribbon is checked, the color and shape specified on the Clusters worksheet will be shown on the graph in place of any color or shape information found in the Vertices worksheet. Information in the Vertices worksheet is not overwritten, it is simply not displayed. Un-checking the box will display the color and shape information on the Vertices worksheet instead of clusters, but for now leave the box checked so you can see the effect of your newly created clusters on the graph.

![Figure 40. Clusters worksheet](image)

Refresh the graph to see a graph that looks something like Figure 41. You’ll notice a clear clustering between the Republican and Democratic Senators, as well as the tendency of the independent senators to vote Democratic.

![Figure 41. 2007 Senate co-voting network showing Republicans (Red), Democrats (Blue), and Independents (Yellow)](image)
Changing Advanced Layout Options:
You may have noticed the senators in Figure 41 are more spread out than those in Figure 38. NodeXL allows you to change the parameters (i.e., settings) for the Fruchterman-Reingold layout to make vertices spread out or move closer together. To change this setting, go to the Options dialogue box above the graph pane and select the “Layout…” button in the bottom-right corner. This will open up the Layout Options dialogue box shown in Figure 42. Increase the Strength of the repulsive force between vertices to 8.0 and click OK. Clicking on Lay Out Again in the graph pane will show the resulting graph more similar to Figure 41 than Figure 38.

Creating Clusters Automatically:
NodeXL includes the capability to automatically identify clusters. Currently the algorithm described in the article "Finding Community Structure in Mega-scale Social Networks" by Ken Wakita and Toshiyuki Tsurumi is used to create clusters. Click on the Find Clusters button in the Analysis section of the NodeXL ribbon. This will replace the data you manually entered on the Clusters and Cluster Vertices worksheets with the automatically generated clusters. Each cluster is given a numerical ID that is shown in Column A of both worksheets (e.g., see Figure 39). Colors and Shapes are automatically assigned to each cluster (Figure 43).
Go to the Cluster Vertices worksheet to see which vertices are assigned to which cluster. To view the results, you’ll need to make sure the Clusters box is checked in the Show/Hide portion of the NodeXL ribbon and Refresh the graph. Figure 43 shows the result. The graph shows that the clustering algorithm was able to identify the two most distinct groups, although the automatically assigned colors are not what people would expect (i.e., the Republican cluster is now blue and the Democratic cluster is now yellow). You can fix these colors by choosing more appropriate ones from the drop-down menu in the Vertex Color column on the Clusters worksheet (see Figure 43).

There are also differences in which cluster some of the individuals were assigned to. The automatic algorithm created a single person cluster (Collins) because he did not fit well into either of the other clusters (although he considers himself a Republican). The algorithm also grouped Snowe and the two independent Senators (Lieberman and Sanders) in the Democratic cluster even though they are not technically Democrats. The number of clusters is not predetermined (i.e., it won’t always be 3). Likewise the number of vertices in each cluster can vary significantly.

![Figure 44. Automatically generated clusters showing 3 unique clusters](image)

**Showing and Hiding Clusters:**
You don’t need to show the cluster information on the graph. To hide this information from the graph, uncheck the Clusters box in the Show/Hide section of the NodeXL ribbon and refresh the workbook. The clustering information will be retained on the Cluster worksheets, but the visual display will be determined by what is on the Vertices worksheet. In this case it will revert back to looking like Figure 38.

You may want to experiment with the dataset to practice some of the features introduced earlier. For example, you can calculate metrics to find the individuals with the highest Betweenness Centrality or other metrics of interest. You could also adjust the Edge Visibility option to a different number (e.g., 50% agreement) or use the dynamic filters to see how the network changes as the variables are modified.
9) Conclusion

This taste of social media network analysis is just the beginning. You have now used the main features of NodeXL. However, knowing how to apply them effectively to solve real-world problems can take a significant amount of practice. As you tackle increasingly complex datasets and challenging problems, you’ll be contributing to the lively and growing field of social network analysis.
10) References and resources


