discourseGT: An R package to analyze discourse networks in educational contexts

Joshua P. Le
University of California
San Diego

Albert Chai
University of California
San Diego

Andrew S. Lee
University of California
Los Angeles

Kevin Banh
University of California
San Diego

Priya Pahal
University of California
San Diego

Stanley M. Lo
University of California
San Diego

Abstract

Student collaborations and discussions in the classroom are important components of the learning process. Current methods for analyzing discourse in these educational contexts are predominantly qualitative. To complement the existing methodologies in education research, discourseGT is an R package that adapts graph theory and social network analysis to analyze conversation patterns when students collaborate in small groups. This software package takes data on the sequential student talk-turns in a classroom discussion and produces statistics and plots based on both graph theory and other non-graph theory parameters. Overall, these new features in discourseGT can provide insight on the dynamics of student discussions relevant to education researchers.

Keywords: student group work, discourse network, graph theory, discourseGT, R.

1. Introduction

Vygotsky postulated that social interactions are crucial to individual learning and cognitive development (Vygotsky 1978). Subsequent studies in education research have found that small-group discussions can help students develop disciplinary understanding (Freeman, Eddy, McDonough, Smith, Okoroafor, Jordt, and Wenderoth 2014) and enhance skills in problem solving (Heller, Keith, and Anderson 1992), critical thinking (Webb 1982, Gokhale (1995), Bligh, McNay, and Thomas (2000)), communication (Webb and Farivar 1994), and metacognition (Webb and Mastergeorge 2003, Veenman, Hout-Wolters, and Afferbach (2006), Bromme, Pieschl, and Stahl (2010)). Furthermore, small-group discussions can also improve student interest and motivation (Skinner and Belmont 1993, Ryan (2000)) as well as course completion rates and persistence in science, technology, engineering, and mathematics (STEM) majors
**discourseGT**: An R package to analyze educational discourse networks


To complement these existing methodologies, we have developed a process that considers student discourse as a network (a “discourse network”) and quantitatively examines the dynamics of small-group discussions using graph theory and network analysis (Chai, Lee, and Lo 2019, Liyanage, Lo, and Hunnicutt (2021)). Many contemporary applications and software packages are optimized for large-scale networks. For example, igraph (Csardi and Nepusz 2006), network (Butts 2021), and sna (Butts 2020) were developed to analyze social media networks (Jones, Bond, Bakshy, Eckles, and Fowler 2017), epidemiological networks (Christakis and Fowler 2011), and political networks (Hobbs, Burke, Christakis, and Fowler 2016), respectively. In contrast, discourse networks in educational contexts are substantially smaller, typically with only 3-8 students (Wagner and González-Howard 2018). Consequently, certain parameters that are relevant for these larger networks are not necessarily applicable, and analysis of discourse networks further demands additional parameters beyond what is available in graph theory (Chai et al. 2019, Lou, Abrami, and d’Apollonia (2001)).

We have decided to develop our software package with the R programming language (R Core Team 2021) because of its open-source nature and extensibility of packages. Instead of rebuilding every component from the beginning, we use existing network analysis software packages, including igraph (Csardi and Nepusz 2006), ggp saying (Kassambara 2020), G-Gally (Schloerke, Crowley, Cook, Briatte, Marbach, Thoen, Elberg, and Larmarange 2021), network (Butts 2021), ggplot2 (Wickham, Chang, Henry, Pedersen, Takahashi, Wilke, and Woo 2021a), dplyr (Wickham, François, Henry, and Möller 2021b), ggrepel (Slawikowski 2021), and sna (Butts 2020), as well as other relevant software packages such as BiocManager (Morgan and Ramos 2021) and RCy3 (Pico, Muetze, Shannon, Isserlin, Pai, Gustavsen, and Kolishovski 2021) in R, and Cytoscape (Shannon, Markiel, Ozier, Baliga, Wang, Ramage, Amin, Schwikowski, and Ideker 2003). At the time of writing, the current version of the package is 1.1.7 (https://CRAN.R-project.org/package=discourseGT) (Chai, Lee, Le, and Lo 2021), and the current version of the graphical user interface (GUI) is 1.1.0 (https://sites.google.com/ucsd.edu/dgt/home).

This paper is organized into five additional sections beyond the Introduction. Section 2 describes important background information including an operationalized definition of discourse networks with relevant graph theory parameters and other non-graph theory parameters.
Section 3 outlines the general workflow of discourseGT. Section 4 offers a step-by-step case example that contextualizes the workflow with sample data. Section 5 examines the potential limitations and future developments of discourseGT. Section 6 provides information on the software package and dependency versions used to generate the results in this paper. Section 7 acknowledges other contributors to the development and distribution of this software package.

2. Background

In any network, there exist nodes with edges connecting them (Godsil and Royle 2001). The precise meaning for these nodes and edges can change depending on the context of the network. Discourse networks require relational data that join two participants in a discussion with some kind of discourse connection (Wagner and González-Howard 2018). In our discourseGT, we track the sequential order of students who speak in a small-group discussion (Chai et al. 2019). Nodes represent members of the group, which are students and can also sometimes include peer facilitators, teaching assistants, and/or the instructor of a course. Edges represent talk-turns or the progression of different individuals speaking. A directed edge pointing from Node A to Node B indicates that Participant B spoke after Participant A. This directionality does not necessarily indicate that Participant A talked directly to Participant B, as all members of the group likely listened to the conversation. Instead, edges could be interpreted in a few different ways. Beyond tracking the sequential order of talk-turns, edges may indicate who is willing to speak after others, who contributes ideas that could be expanded upon or responded to, and/or who has the agency or power at the moment to speak in the group (Chai et al. 2019).

2.1. Graph Theory Parameters

In our previous work (Chai et al. 2019, Liyanage et al. 2021), we identified a subset of graph theory parameters that are relevant to small-group discussions in educational contexts as shown in Table 1.
Table 1: Graph theory parameters used in discourseGT

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Graph Theory Definition</th>
<th>Social Network Definition</th>
<th>Discourse Network Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>An object of interest</td>
<td>Typically a person</td>
<td>Participant in a small group discussion</td>
</tr>
<tr>
<td>Edge</td>
<td>A connector between two nodes</td>
<td>Flow of information between two people</td>
<td>Talk-turn between two individuals</td>
</tr>
<tr>
<td>Direction</td>
<td>Defines which node points to another using the edge</td>
<td>Indicates which person has ties to the other</td>
<td>Indicates which individual talks after the other</td>
</tr>
<tr>
<td>Weight</td>
<td>A number associated with an edge</td>
<td>Frequency of information flow between two people</td>
<td>Frequency of a talk-turn between two individuals</td>
</tr>
<tr>
<td>Degree</td>
<td>Number of edges connected to a node</td>
<td>Number of people an individual has ties to</td>
<td>Number of people an individual talks before and after</td>
</tr>
<tr>
<td>Density</td>
<td>Number of edges divided by the number of possible edges</td>
<td>Interactions occurring among different people</td>
<td>Talk-turns occurring among different individuals</td>
</tr>
<tr>
<td>Centrality</td>
<td>A number for the importance of a given node in the graph</td>
<td>Amount of influence of each person</td>
<td>Amount of talk-turn contribution of an individual</td>
</tr>
<tr>
<td>Centralization</td>
<td>A number for the importance of the central node</td>
<td>Dependence of a network on its most active person</td>
<td>Dependence of a group on its most active individual</td>
</tr>
<tr>
<td>Subgraph</td>
<td>A smaller graph within a graph</td>
<td>Individuals who have closer ties to each other</td>
<td>Individuals who talk after each other more</td>
</tr>
</tbody>
</table>

2.2. Non-Graph Theory Parameters

In addition to parameters derived from graph theory and network analysis, discourseGT also computes non-graph theory (NGT) parameters that can provide additional insight into the dynamics of student discussions in small groups. These include episode length, number of talk-turns per hour, normalized turn ratio (NTR), and an equitability measurement.

In discourse networks, an “Episode” describes a subset of talk-turns within a discussion. The beginning and end of an episode are defined by and dependent on the researcher’s interest — e.g. bounded by responses to a specific question (Chai et al. 2019) or talk-turns within a specific classroom activity (Liyanage et al. 2021). Episode length is the number of talk-turns within an episode, e.g. from question to question or from activity to activity. Longer episodes may indicate that participants are engaged in more in-depth discussions that require multiple talk-turns to elaborate on ideas.

The number of talk-turns per hour describes the rate at which a participant contributes to the conversation, allowing for the direct comparison of all participants within a group.
In discourseGT, a talk-turn can be classified as either a start to an episode (\textit{ep\_start}) or a continuation of an episode (\textit{ep\_cont}). Therefore, discourseGT computes the number of episode-start talk-turns per hour (Equation 1) and the number of episode-continue talk-turns per hour (Equation 2).

\[
\text{ep\_start per hour} = \frac{\text{Number of episode starts by a participant}}{\text{Time in hours}} \tag{1}
\]

\[
\text{ep\_cont per hour} = \frac{\text{Number of episode continuations by a participant}}{\text{Time in hours}} \tag{2}
\]

Normalized turn ratio (NTR) allows for the comparison of participant activities across discourse networks that may have different total numbers of talk-turns (Chai et al. 2019). In a given discussion, discourseGT calculates the fair-share number of talk-turns per participant (Equation 3), and NTR describes each of their participation relative to this fair share (Equation 4). If a participant has an NTR >1.0, then they talked more than their fair share of talk-turns, independent of the size of the group. Similarly, if a participant has an NTR of <1.0, then they talked less than their fair share of talk-turns.

\[
\text{Fair share number of talk-turns} = \frac{\text{Total number of talk-turns}}{\text{Number of group participants (nodes)}} \tag{3}
\]

\[
\text{Normalized Turn Ratio (NTR)} = \frac{\text{Number of talk-turns by a participant}}{\text{Fair share number of talk-turns}} \tag{4}
\]

To determine the equitability of talk-turn contributions from different participants within a group, discourseGT uses the Shannon Evenness Index ($E_H$). $E_H$ relies on the Shannon Diversity Index ($H'$), which was originally developed to measure the diversity or uncertainty of words in a string of text (Shannon 1948) and has been widely used as a measurement of biodiversity within ecosystems (Pielou 1966). $E_H$ allows for the direct comparison of discourse networks among each other with a single measurement that describes the equitability of the distribution of talk-turns among all participants within a small-group discussion.

\[
E_H = \frac{H'}{\ln S} \tag{5}
\]

\[
H' = - \sum_{i=1}^{S} p_i \cdot \ln p_i \tag{6}
\]

Here, $i$ is the index for each individual node representing a participant, and $S$ is the total number of participants in a discourse network. For each node, $p_i$ describes the proportion of talk-turns taken by that participant out of the total number of talk-turns in the discourse network.
$E_H$ has a range from 0 to 1, inclusive, where 1 represents the maximum equitability among the participants within the discourse network.

3. discourseGT Workflow

3.1. General Workflow

The functions of discourseGT were designed to be as modular as possible, making it possible to only run analyses of interest. Figure 1 represents the general workflow of discourseGT, and Table 2 describes explicit function names organized by their general uses.

Figure 1: General workflow of discourseGT. The raw data can either be converted to an igraph object for further analysis or directly passed for NGT analysis. All console output can be permanently stored to the user’s local disk. Green represents the start of the workflow. Purple represents steps necessary to generate an igraph object. Blue represents the potential downstream uses of an igraph object. Orange represents NGT analysis. Red signals the end of the workflow.
Table 2: List of all discourse\textbf{GT} functions

<table>
<thead>
<tr>
<th>Phase</th>
<th>Function Name</th>
<th>Parameter(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purple</td>
<td>tabulate_edges()</td>
<td>input – data.frame or string. Points to .csv file with talk-turn data in the question-and-response format. iscsvfile – boolean. TRUE if input is a .csv file. Else FALSE. silentNodes – integer. The number of nodes that do not interact with others.</td>
<td>Calculates the weighted edge list from the input data and number of silent nodes not captured in the data.</td>
</tr>
<tr>
<td>Purple</td>
<td>prepareGraphs()</td>
<td>raw_data_input – list. Output of tabulate_edges(). project_title – string. Sets the title of the project. weightedGraph – boolean. TRUE if downstream analysis should account for weighted edges. Else FALSE.</td>
<td>Prepares the igraph object from the weighted edge list. This is utilized by several downstream analytical functions.</td>
</tr>
<tr>
<td>Blue</td>
<td>coreNetAnalysis()</td>
<td>ginp – list. Output of prepareGraphs().</td>
<td>Analyzes the input igraph object and returns basic network statistics, as reasoned in Chai et al. 2019.</td>
</tr>
<tr>
<td>Phase</td>
<td>Function Name</td>
<td>Parameter(s)</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>----------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Blue</td>
<td>subgroupsNetAnalysis()</td>
<td>ginp - list. Output of prepareGraphs().</td>
<td>Analyzes the input igraph object for potential subgroups.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>raw_input - data.frame. Points to the original talk-turn data in the question-and-response format.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>normalized - boolean. Whether or not to normalize the betweenness centrality values relative to the graph.</td>
<td></td>
</tr>
<tr>
<td>Blue</td>
<td>summaryNet()</td>
<td>netintconfigData - list. Output of prepareGraphs().</td>
<td>Summarizes the analytical output from several other functions into a single output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>coreNetAnalysisData - list. Output of coreNetAnalysis().</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>subgroupsNetAnalysisData - list. Output of subgroupsNetAnalysis().</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>display - boolean. Whether or not to print output to console.</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>Function Name</td>
<td>Parameter(s)</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Blue</td>
<td>basicPlot()</td>
<td>ginp – list. Output of prepareGraphs().</td>
<td>Plots a basic network graph utilizing the default R visualization backend.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>graph_selection_input – integer. Numerical value from 0 to 2, inclusive, which selects the graphing algorithm used. 0 = Fruchterman Reingold, 1 = Kamada Kawai, and 2 = Reingold Tilford.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>curvedEdgeLines – boolean. Whether or not to curve graph edges.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>arrowSizeMultiplier – numeric. Scales arrow sizes based on input factor.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logscale – boolean. If TRUE, scale graph edges logarithmically. Else do not.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logBase – integer. Logarithmic base to scale graph edges.</td>
<td></td>
</tr>
</tbody>
</table>
### discourseGT: An R package to analyze educational discourse networks

<table>
<thead>
<tr>
<th>Phase</th>
<th>Function Name</th>
<th>Parameter(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue</td>
<td>plot1Att()</td>
<td>data – list. Output of prepareGraphs().</td>
<td>Plots a network graph with a single input attribute. Utilizes the ggplot2 [@R-ggplot2] backend.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>prop – integer. Rescales the graph edge sizes.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>graphmode – string. Specifies the graphing algorithm used. Refer to gplot.layout for more options.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>attribute – list. Mapping to the attribute information.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>attribute.label – string. Name of attribute to display in the graph.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>attribute.node.labels – list. Mapping to the node labels.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>attribute.nodesize – integer or list. Mapping to universal or individualized node sizes, respectively.</td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>Function Name</td>
<td>Parameter(s)</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Blue</td>
<td>plot2Att()</td>
<td>data – list. Output of prepareGraphs(). prop – integer. Rescales the graph edge sizes. graphmode – string. Specifies the graphing algorithm used. Refer to gplot.layout for more options. attribute1 – list. Mapping to the first attribute information. attribute2 – list. Mapping to the second attribute information. attribute1.label – string. Name of the first attribute to display in the graph. attribute2.label – string. Name of the second attribute to display in the graph. attribute.node.labels – list. Mapping to the node labels. attribute.nodesize – integer or list. Mapping to universal or individualized node sizes, respectively.</td>
<td>Plots a network graph with two input attributes. Utilizes the ggplot2 [@R-ggplot2] backend.</td>
</tr>
</tbody>
</table>
**discourseGT**: An R package to analyze educational discourse networks

<table>
<thead>
<tr>
<th>Phase</th>
<th>Function Name</th>
<th>Parameter(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orange</td>
<td>plotNGTData()</td>
<td>data: data.frame or string.</td>
<td>Analyzes non-graph theory statistics and visualizes them in three plots. These are elaborated on in Chai et al. 2019.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>convMinutes: integer.</td>
<td>Length of conversation, in minutes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>iscsvfile: boolean.</td>
<td>TRUE if input is a .csv file. Else FALSE.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>silentNode: integer.</td>
<td>The number of nodes that do not interact with others.</td>
</tr>
<tr>
<td>Red</td>
<td>writeData()</td>
<td>project_name: string.</td>
<td>Writes any data object file as an appropriate format to a specified user directory. Images are saved with a resolution of 300dpi.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>objectfile: list.</td>
<td>The object to be exported to disk.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dirpath: string.</td>
<td>The location on disk where the exported file will be written.</td>
</tr>
</tbody>
</table>

### 3.2. Data Structure

Collecting and formatting data for analysis by discourseGT is based on episodes and talk-turns (Chai et al. 2019). Talk-turn data should be recorded as participants speak sequentially, which can be done with life observations in real time (Chai et al. 2019) or analysis of video or audio transcripts (Liyanage et al. 2021). Be prepared to record the duration of the discussion (in minutes), which is required to determine the number of episode starts and episode continuations per unit of time. Talk-turn data are collected in a two-column table that tracks episode starts (ep_start) and episode continuations (ep_cont) and with each participant in the group assigned a unique identifier, such as a number (Table 3). Each row should only have a single participant’s identifier entered once either in the ep_start or ep_cont column.
An entry in the `ep_start` column denotes the beginning of a new episode. The boundaries of an episode are defined by the researcher and the research question, although these definitions should be set consistently within a study. It is vital that the column names in the data are explicitly labeled as `ep_start` and `ep_cont`, respectively. Raw data may be prepared using most spreadsheet software or text editors, but it should ultimately be saved as a comma-separated file (.csv).

Table 3: Formatted talk-turn data ready for discourseGT analysis. In this example, an episode is defined arbitrarily as a topic (not shown) — that is, each episode is a relevant discussion on a single topic. There are two episodes. The first episode is three talk-turns long, with Participant 1 initiating the episode. Participant 3 then spoke, followed by Participant 2. The second episode has two talk-turns, with Participant 4 starting a new episode and Participant 2 speaking next to complete the overall discussion. It is important to note that the duration of the conversation (in minutes) is not a part of the table. Rather, it should be recorded elsewhere for use in NGT analysis.

<table>
<thead>
<tr>
<th>ep_start</th>
<th>ep_cont</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NA</td>
</tr>
<tr>
<td>NA</td>
<td>3</td>
</tr>
<tr>
<td>NA</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>NA</td>
</tr>
<tr>
<td>NA</td>
<td>2</td>
</tr>
</tbody>
</table>

4. Worked Case Example

The discourseGT software package comes equipped with example data. Here, we will utilize these data to demonstrate its utility in examining discourse networks.

To get started, install the software package through the Comprehensive R Archive Network (CRAN). Load it using:

```r
R> library(discourseGT)
```

4.1. Importing Data

Raw data can be imported using the `read.csv()` function. For the sake of utilizing the example data, however, it is useful to duplicate it by assigning its values to a new variable. Once it has been duplicated, view the head of the data to ensure that it has been properly imported:

```r
R> data <- sampleData1
R> head(data)

   ep_start ep_cont
1      1     NA
```
4.2. Preparing the igraph Object

Prior to generating the igraph object, a weighted edge list needs to be generated from the imported raw data. By default, the weight of an edge is defined as the number of times an edge has occurred between two nodes. Weights can be redefined based on other available criteria, but this must be done manually.

```r
R> # Calculate the weighted edge list
R> tabEdge <- tabulate_edges(data, iscsvfile = FALSE, silentNodes = 0)
R> # Check the weighted edge list
R> head(tabEdge$master)

source target weight
1 1 1 8
2 2 1 25
3 3 1 49
4 4 1 75
5 1 2 28
6 3 2 11
```

Recall that an igraph object is the core input to many of the modular analytical functions offered in discourseGT. To generate an igraph object, the following information is required:

- The variable that stores the weighted edge list
- The title of the project. Default: null
- Is the graph weighted? Default: TRUE

```r
R> prepNet <- prepareGraphs(tabEdge, project_title = "Sample Data 1", +     weightedGraph = TRUE)
```

The graph settings specified by this function will influence the analytical output of downstream functions.

4.3. Running Graph Theory Analysis

discourseGT offers graph theory-based analytics via two separate functions. The first, `coreNetAnalysis()`, will perform core operations that produce the parameters previously detailed in Table 1 on page 4. It will count the number of nodes, and edges, calculate edge weights, average graph degree, modularity, centrality, and related graph theory parameters. To run the function and store it in a variable:
The second, `subgroupsNetAnalysis()`, utilizes the Girvan-Newman algorithm to detect subgroups within the overall network (Girvan and Newman 2002), such that:

```r
R> subNet <- subgroupsNetAnalysis(prepNet, raw_input = data, normalized = TRUE)
```

### 4.4. Generating Summaries

While it is possible to display the generated `igraph` object, core network statistics, and subgroup statistics as separate outputs, it can be helpful to view them as an overall summary of a network’s graph theory analytics. Furthermore, combining all of these outputs into a single variable is a necessary step in exporting them as a single text file. The `summaryNet()` function will combine the outputs from `prepareGraphs()`, `coreNetAnalysis()`, and `subgroupsNetAnalysis()` as such:

```r
R> summaryData <- summaryNet(netintconfigData = prepNet, coreNetAnalysisData = coreNet, + subgroupsNetAnalysisData = subNet, display = TRUE)
```

```
BEGIN SUMMARY

discourseGT R Package - Production
Package Version: [1] '1.1.7'
Graph Results - Project Summary

------------PROJECT DETAILS------------
Name of Project: Sample Data 1
Summary Results Generated On: [1] "2021-09-03 03:17:51 EDT"

------------GRAPH CONFIGURATION------------
Weighted Graph: TRUE

**********CORE PARAMETERS ANALYSIS**********
Number of Edges: 12
Number of Nodes: 4
Weighted Edges: 465
Graph Adjacency Matrix:
4 x 4 sparse Matrix of class "dgCMatrix"
1 28 47 74
2 25 13 14
3 49 11 . 52
4 75 13 52 .

Network Density: 1
Average Degree: 6
Strong/Weak Interactions:
```
1 2 3 4
1 1 1 1

Unrestricted Modularity:

---------------GRAPH CENTRALITY---------------
Degree Centrality:
$\text{res}$
[1] 6 6 6 6
$\text{centralization}$
[1] 0
$\text{theoretical_max}$
[1] 12

Articulation Points List:
+ 0/4 vertices, named, from 0cb105f:

Reciprocity: 1

---------------SUBGROUPS AND MODULARITY---------------
Girvan-Newman Subgroups Detection:
IGRAPH clustering edge betweenness, groups: 1, mod: 0
+ groups:
  $`1`$
  [1] "1" "2" "3" "4"

Betweeness:
1 2 3 4
0 1 0 0

Normalized Betweenness: TRUE

Group Core Members:
1 2 3 4
6 6 6 6

Graph Symmetry of Members:
$\text{mut}$
[1] 6
$\text{asym}$
[1] 0
Graph Connectedness Census:

4
1

Neighborhood List for Each Adjacent Node:
[[1]]
+ 4/4 vertices, named, from 0cb105f:
[1] 1 2 3 4

[[2]]
+ 4/4 vertices, named, from 0cb105f:
[1] 2 1 3 4

[[3]]
+ 4/4 vertices, named, from 0cb105f:
[1] 3 1 2 4

[[4]]
+ 4/4 vertices, named, from 0cb105f:
[1] 4 1 2 3

Transitivity/Clustering Coefficients:
Local Transitivity values:
[1] 0.2 0.2 0.2 0.2
Global Transitivity values:
[1] 1

---------DISCLAIMER AND WARRANTY OF PROVIDED RESULTS AND CODE---------
Results from Code:
The researcher(s) are primary responsible for the
interpretation of the results presented here with the script.
The authors accept no liability for any errors that
may result in the processing or the interpretation of
your results. However, if you do encounter errors in
the package that should not have happened, please let us
know

Code Warranty:
MIT License
Copyright (c) 2018 Albert Chai, Andrew S. Lee, Joshua P. Le, and Stanley M. Lo
Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the 'Software'), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED 'AS IS', WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

4.5. Basic Visualization

discourseGT offers several methods to visualize networks. For a basic network graph, basicPlot() should be used, and its parameters should be modified to suit the needs of the user. These options include modifications to the plotting algorithm, edge curvature, arrow size, and edge weight scaling.

Its default plotting algorithm is Fruchterman Reingold, denoted by 0 (Fruchterman and Reingold 1991). This is typically the best option to use because it attempts to minimize edge intersections in the final plot, improving readability. Other projections include Kamada Kawai (Kamada and Kawai 1989) and Reingold Tilford (Reingold and Tilford 1981), denoted by 1 and 2, respectively.

Edge curvature defaults to TRUE so that differences in talk-turn taking between nodes can be distinguished. Consider two participants, represented as Node A and Node B. It is entirely possible for Node A to talk after Node B more than Node B talks after Node A. Consequently, the two edges that point in each direction will have different weights, and these can only be visually seen if they are curved instead of overlapping. On the other hand, graphs without curved edges may improve clarity. This can be especially favorable when plotting an unweighted graph.

To modify arrow sizes, a multiplier can be passed to arrowSizeMultiplier. The default value is 1. Any values <1.0 will shrink the arrow, and vice versa. Again, this feature is added to improve readability in specific cases.

Lastly, edge weight scaling is best used for improved visualization of larger, weighted datasets. Due to the increase in raw edges, default plotting may yield unreadable results. We implemented Equation 7 to do so according to a linear scale. This method allows for users to
visually compare talk-turn frequencies within a graph, which is not as intuitive with other forms of scaling.

$$y = \frac{(scaledMax - scaledMin) \cdot (eachEdgeWeight - rawMin)}{rawMax - rawMin} + scaledMin$$  \hspace{1cm} (7)

Here, each edge weight is individually scaled to a new value $y$. $scaledMax$ and $scaledMin$ are the user-defined boundaries of a new scale for all weighted edges. $rawMin$ and $rawMax$ are the minimum and maximum edge weights that are extracted from the raw data via the `prepareGraphs()` function. $eachEdgeWeight$ refers to the weight of each unique edge.

For users, $scaledMax$ must be greater than or equal to $scaledMin$. These variables may also be set to equal, non-zero values to produce an unweighted version of the graph.

Note that while both $scaledMin$ and $scaledMax$ can theoretically be set to 0, we advise against this because the resulting graph will appear to have no edges. Likewise, if $scaledMin$ is set to 0 while $scaledMax$ is a non-zero value, the resulting graph will appear to have no edges where the most infrequent talk-turns occurred. This may have some functionality depending on the user’s use-case.

Below is an example of a graph that uses the Fruchterman Reingold projection, linearly scales the dataset to new weighted edge boundaries of $[1, 10]$, and applies a scale of 2 to the arrow sizes.

\[
R \texttt{> basicPlot(prepNet, graph_selection_input = 0, curvedEdgeLines = TRUE,}
+ \texttt{ arrowSizeMultiplier = 2, scaledEdgeLines = TRUE, scaledMin = 1,}
+ \texttt{ scaledMax = 10)}
\]
Sample Data 1

In this plot, it can be easily seen that the fewest number of talk-turns relative to the entire discourse network occurred between Nodes 2 and 3 as well as Nodes 2 and 4. Nodes 1 and 2 shared the next fewest number of talk-turns, followed by Nodes 1 and 3 and Nodes 3 and 4. Nodes 1 and 4 shared the greatest number of talk-turns between them. In each of these node pairs, the conversation appeared to travel equally between the nodes involved, as the edges of similar thickness indicate. Note that we cannot view any attribute data about the nodes here.

4.6. Attribute Visualization

To add attributes to a network graph, the `plot1Att()` and `plot2Att()` functions can be used. These functions utilize the `ggplot2` backend with `GGally` (Wickham et al. 2021a, Schloerke et al. (2021)), giving them an appearance distinct from the previously discussed `basicPlot()` function.

Before starting, ensure that a properly formatted `data.frame` with attributes is in the working environment. Displayed below is an example attribute dataset included with `discourseGT`:

```r
R> attData <- attributeData
R> head(attData)
```
Note that the first column, node, contains each node name that was included in the initial imported data. This is a crucial aspect to the attribute data because it identifies attributes associated with particular nodes for plot1Att() and/or plot2Att().

Similarly to the basicPlot() function, the attribute plotting functions include options to modify the overall projection, albeit less granular. These include edge scaling, node sizes, and plotting algorithm.

Edge weight scaling can be modified by changing the value of prop, and node sizes can be modified by changing the value of attribute.nodesize. Each of these have a default value of 20, although this is arbitrary. The user should find the best settings that suit their use case.

The default plotting algorithm is again Fruchterman Reingold for its readability (Fruchterman and Reingold 1991). Here, however, this option is indicated by passing fruchtermanreingold into the function. Other projections can be found with gplot.layout.

Lastly, it is important to note that only 1 or 2 attributes can be plotted at once. These cases should utilize the plot1Att() and plot2Att() functions, respectively.

Below is an example of an attribute graph with larger-than-default edge sizes and smaller-than-default node sizes. It utilizes the Fruchterman Reingold projection.

```R
R> plot1Att(prepNet, +   prop = 40, +   graphmode = "fruchtermanreingold", +   attribute = attData$ethnicity, +   attribute.label = "Ethnicity", +   attribute.node.labels = attData$node, +   attribute.nodesize = 16)

Registered S3 method overwritten by 'GGally':
    method from
    *.gg  ggplot2
```
**discourseGT**: An R package to analyze educational discourse networks

```r
$g2plot

Sample Data 1

To plot a second attribute to a network, utilize `plot2Att()` with the aforementioned notation. The following graph showcases the network with both ethnic and gender data:

```r
R> plot2Att(prepNet,
+          prop = 40,
+          graphmode = "fruchtermanreingold",
+          attribute1 = attData$ethnicity,
+          attribute2 = attData$gender,
+          attribute1.label = "Ethnicity",
+          attribute2.label = "Gender",
+          attribute.node.labels = attData$node,
+          attribute.nodesize = 16)

$g2plot
```
Sample Data 1

4.7. Customizable Visualization

Further graph customizability, such as node placements, can be achieved with Cytoscape, an open-source network plotting software (Shannon et al. 2003). In order to utilize this method:

1. Download & install Cytoscape.
3. Plot the igraph object and modify it in Cytoscape.

Assuming that Cytoscape is installed, install and load RCy3 to properly link it to R. This can be done by:

```r
R> install.packages("BiocManager")
R> BiocManager::install("RCy3")
R> library(RCy3)
```
To plot a graph, first ensure that a new Cytoscape session is loaded. Then, utilize the following command to send an igraph object to the GUI:

\[ R> \text{createNetworkFromIgraph(prepNet$graph)} \]

The graph will now appear in Cytoscape, where further modifications can be made.

### 4.8. Running Non-Graph Theory Analysis

Recall that discourseGT does not require an igraph object to produce NGT analysis. Rather, plotNGTData() utilizes the raw, two column data to generate its output. Additionally, it requires the duration of the conversation (in minutes) and the number of silent nodes (i.e. participants who did not speak at all) in the discourse network. The function outputs the previously-discussed NGT parameters and three individual graphs. The raw data are also exported alongside the graphs, giving the user greater flexibility in creating their own NGT visualizations.

\[ R> \text{plotNGTData(data = data, convoMinutes = 90, iscsvfile = FALSE, + silentNodes = 0)} \]
### $ngt_std_stats1$

<table>
<thead>
<tr>
<th>participant</th>
<th>ep_start</th>
<th>ep_cont</th>
<th>total_count</th>
<th>total_edges_in_out</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>27</td>
<td>131</td>
<td>158</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>6</td>
<td>46</td>
<td>52</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>11</td>
<td>104</td>
<td>115</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>20</td>
<td>121</td>
<td>141</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>edge_by_part</th>
<th>ep_starts_hour</th>
<th>ep_conts_hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>157</td>
<td>18.000000</td>
</tr>
<tr>
<td>2</td>
<td>52</td>
<td>4.000000</td>
</tr>
<tr>
<td>3</td>
<td>115</td>
<td>7.333333</td>
</tr>
<tr>
<td>4</td>
<td>141</td>
<td>13.333333</td>
</tr>
</tbody>
</table>

### $ngt_std_stats2$

<table>
<thead>
<tr>
<th>length_of_ep</th>
<th>freq_of_ep</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td>15</td>
<td>18</td>
</tr>
<tr>
<td>16</td>
<td>20</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>13</td>
</tr>
<tr>
<td>19</td>
<td>17</td>
</tr>
<tr>
<td>20</td>
<td>19</td>
</tr>
</tbody>
</table>

### $ngt_adv_stats$

<table>
<thead>
<tr>
<th>participant</th>
<th>normalized_turn_ratio</th>
<th>indv SDI_arg</th>
<th>SDI</th>
<th>SEI</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.3505376</td>
<td>-0.3666006</td>
<td>1.318946</td>
<td>0.9514183</td>
</tr>
<tr>
<td>2</td>
<td>0.4473118</td>
<td>-0.2449920</td>
<td>1.318946</td>
<td>0.9514183</td>
</tr>
<tr>
<td>3</td>
<td>0.9892473</td>
<td>-0.3455207</td>
<td>1.318946</td>
<td>0.9514183</td>
</tr>
<tr>
<td>4</td>
<td>1.2129032</td>
<td>-0.3618325</td>
<td>1.318946</td>
<td>0.9514183</td>
</tr>
</tbody>
</table>

### $episodes_plot$
discourseGT: An R package to analyze educational discourse networks
4.9. Exporting to Disk

The `writeData()` function accepts specific `discourseGT` function output and exports it as a permanent file to a specified directory on the user's disk. It can save the generated summary object, any plots, and weighted edge lists. Images will automatically export as a `.tiff` at 300 DPI, and console output will be exported as a `.txt` file.

The following example exports the generated summary to disk:

```R
writeData("Sample Data 1", summaryData, dirpath = tempdir())
```

5. Discussion and Future Development

This paper demonstrates the workflow of `discourseGT` and introduces basic elements of its underlying methodology and mathematics. Although graph theory parameters, visualization, and non-graph theory parameters relevant to educational contexts are covered, it is important to note some limitations of this package.

Firstly, we use degree centrality as a way of describing node centrality. This is the simplest calculation of centrality and shows the number of direct and distinct talk-turn connections a node has to others in the network. However, this does not give a complete picture of a node’s influence on a network because it only counts the number of distinct connections to other nodes, regardless of their weight. Therefore, this calculation should be interpreted
in conjunction with betweenness centrality and other individual node statistics to provide a broader view of a specific node’s influence on the discourse network.

Secondly, talk-turns do not capture direct interactions between participants. Instead, they only represent the progression of the conversation to track dynamics of the discourse network. This was implemented because in discourse networks, participants can either address the entire group, a subgroup of the network, or another individual, and it is not feasible to distinguish these. Moreover, modeling direct interactions would result in a saturation of of edges in the network. This would not likely yield much useful information.

Thirdly, talk-turns do not capture the quality or quantity of the conversation. In this context, quality refers to the words that are actually spoken, which may be on-or-off-topic. Quantity refers to the length of an interaction, which may be a few words or several sentences long. Our methodology models all talk-turns the same way, regardless of content or length. This was intentionally designed to complement existing methodologies in discourse analysis that typically focus on the content of the discussion (Barros and Verdejo 2000).

Fourthly, discourseGT relies on existing R packages and software for plotting. This construction restricts some meaningful functionality at the discretion of the underlying packages. For example, the attribute plotting functions, plot1Att() and plot2Att(), are built on the ggplot2 (Wickham et al. 2021a) R package. While this allows users to apply ggplot2’s (Wickham et al. 2021a) advanced formatting functionality to their plots, it also prevents the edges from being curved. This is an important feature available in basicPlot(), which is built on the default R plotting backend, that visually distinguishes two edges of different weights that point in opposite directions between a pair of nodes. Consequently, some talk-turn information is lost under the thicker edge.

Fifthly, this dependence on other R packages and software for plotting means that this functionality is susceptible to their version limitations. Most notably, our testing of discourseGT revealed that not all versions of Cytoscape (Shannon et al. 2003) connected properly with the R environment. Only a specific version worked. Although we did not observe this behavior for other dependencies, our experience with Cytoscape (Shannon et al. 2003) demonstrates that this could potentially affect some functionality.

Sixthly, discourseGT does not currently support in-software manipulation of groups — that is, the ability to separately analyze and visualize a subgroup of a larger group. As demonstrated in (Wagner and González-Howard 2018), this functionality can be useful for discourse networks in educational contexts. At present, it is possible, albeit tedious, to run separate analyses of talk-turn data in the two-column format filtered to subgroups containing only the nodes of interest. Nevertheless, we hope that this current R package can help other researchers to more easily employ quantitative approaches to analyzing discourse networks to complement existing qualitative methodologies.

6. Computational Details

The results in this paper were obtained using R 4.1.0 (R Core Team 2021) and packages discourseGT 1.1.7 (Chai et al. 2021), igraph 1.2.6 (Csardi and Nepusz 2006), ggpubr 0.4.0 (Kassambara 2020), GGally 2.1.2 (Schloerke et al. 2021), network 1.17.1 (Butts 2021), ggplot2 3.3.5 (Wickham et al. 2021a), dplyr 1.0.7 (Wickham et al. 2021b), ggrepel 0.9.1 (Slowikowski 2021), BiocManager 1.30.16 (Morgan and Ramos 2021), and sna 2.6 (Butts 2020). These are
all available from the Comprehensive R Archive Network (CRAN).
For increased customizability of network visualizations, RCy3 2.12.4 (Pico et al. 2021) was obtained from BioConductor, and Cytoscape 3.8.2 (Shannon et al. 2003) was used.

7. Acknowledgements

We thank B. N. Nguyen and K. T. Ly for their contributions to the debugging process of discourseGT. We thank L. Hobbie for his input on several package functions and usability, and we thank S. Lam for her early testing of the package with example datasets. We are grateful for the support and work that CRAN Volunteers put into examining and publishing R packages on the Comprehensive R Archive Network. J. P. Le was an undergraduate researcher at the University of California San Diego.

This material is based upon work supported by the National Science Foundation under Grant No. 2022975. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors declare that there are no conflicts of interest.

References


**discourseGT**: An R package to analyze educational discourse networks


discourseGT: An R package to analyze educational discourse networks


Affiliation:

Stanley M. Lo
University of California
San Diego
9500 Gilman Dr #0355 La Jolla, CA 92093-0355
E-mail: smlo@ucsd.edu