UNIVERSIDAD DE PUERTO RICO
RECINTO UNIVERSITARIO DE MAYAGÜEZ

BEAR

A SOCIAL NETWORK ANALYSIS LANGUAGE

Bárbara Patricia González Rivera
Enrique Antonio Marrero Torres
Alberto Antonio Canela Cubero
Raúl André Vargas Frechel
ICOM 4036 – PROGRAMMING LANGUAGES

Table of Contents

Introduction

BEAR is a programming language focused on Social Network Analysis. Social Network Analysis (SNA for short) is, in its basic form, a type of study of human interaction. This type of analysis has gained momentum in the past decade with the rising popularity of Social Media Platforms. Thus, it has been cited as playing an important role in modern society. With this overwhelming need of a medium that allows an easy way of collecting, analyzing and creating graphs in this society is that BEAR was born.

The abbreviation BEAR stands for the following: Bárbara, Enrique, Alberto, Raúl; These are the member of the BEAR development team. By suggestion of Prof. Wilson Rivera, the group took to SNA functionality in developing their language. Nowadays, companies, business, politicians, and movements make their presence known in social media. With it, they can reach a vast number of users. And, in order to impact those they want to target, they often use SNA to figure out the appropriate way to do so. Curiously, SNA can be applied to various domain applications like history, biology, anthropology, and behavioral science.

In the overall scope of Social Network Analysis, there are two important aspects to evaluate: the general structure of a network (also known as socio-centered perspective), and the composition of local network structures (known as the ego-centered perspective). Both aspects contain key information on any given social group, such as the group's cohesiveness and ability to communicate and exchange with other social groups. Inside of each social group or network, individuals may be found (known as actors). This allows for the potential of studying general human interaction, locating and examining epidemics and evaluating historical events.

Recapitulating, BEAR is focused on social network analysis that hopes to facilitate the way users analyze data for a variety of reasons and be able to display it in different types of graphs. Through it, we can gain a better understanding of how humans interact with each other through different mediums and how such interactions impact society.

Language

Tutorial

You can find a quick-start video guide at the BEAR website[1].
To operate the program you'll need to run the BEAR file in Python.
For example, using GitBash navigate to the project repository and execute **python BEAR.py**
From there, you can use the **help** command to give further details on other commands
Below are the inputs used in the video guide:

```
help
Bear create // test from fileSample.csv \\
Bear display test
Bear create
custom
hungry
END
Bear custom + Wilson
24
Male
N/A
Yes
No
Yes
Bear display custom
Bear custom + Raul
22
Male
17
Yes
No
No
Bear display custom
Bear test - node Jessie
Bear display test
Bear custom operations
1
test
union
Bear display union
exit
```

[1] Can be found at https://gonzaba.github.io/BEAR/#project-demo

Reference Manual

Language Definition:

Tokens

Digit	::=	`0-9`
Character	::=	`a-z | A-Z | _ | .`
ID	::=	`a-z | A-Z | 0-9 | _`
Delimiters	::=	**`//`** `|` **`\\`** `|` **`|`** `|` **`[`** `|` **`]`**

Grammar

Define	::=	**`Bear`** `Function`
Function	::=	`Term`

Term	::=	`Add`
		`| Remove`
		`| Create`
		`| Display`
		`| Graph`
		`| Node`
		`| File`
		`| Operations`

Add	::=	`Graph` **`+ [`** `File` **`]`** `| Graph` **`+`** `Node`
Remove	::=	Graph **-** Node
Create	::=	**`create`** `|` **`create //`** `{ID}*` **`\\`** **`| create //`** `{ID}*` **`from`** `File` **`\\`**
Graph	::=	`{ID}*`
Node	::=	**`node`** `{ID}*`
File	::=	`{ID}*  .  {ID}*`
Operations	::=	`Graph` **`operations`**

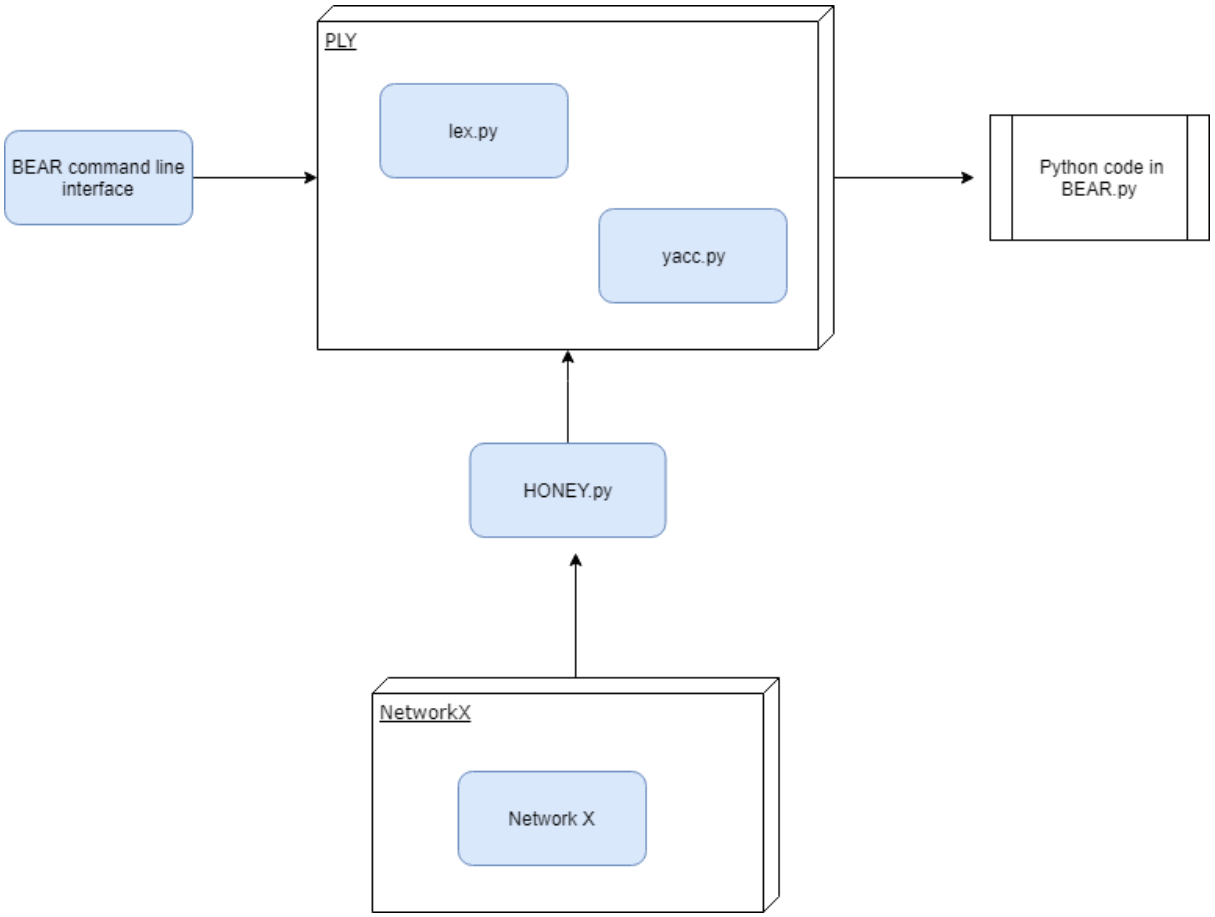
Usable commands:

- Creating networks: `Bear create || Bear create // nameOfGraph \\ ||`
  `Bear create // nameOfGraph from fileName \\`
- Adding Nodes to a Graph: `Bear nameOfGraph + node s ||`
  `Bear nameOfGraph + [ fileName ]`
- Removing Nodes from a Graph: `Bear nameOfGraph - node nameOfNode`
- Displaying a Graph: `Bear display nameOfGraph`
- Graph Operations menu: `Bear nameOfGraph operations`
- Exit BEAR: `exit`

Development

For the core development of this project, we utilized the NetworkX library and their Documentation notes[2].

Translator architecture



The description of the interfaces between the modules goes as it follows:

Upon entering the runtime environment from Bear.py, the system works as an interpreter which for each parsed line of BEAR code calls auxiliary methods from Honey.py. The methods of Honey.py are defined as an interface with the open source library Network X.

The software development environment used to create the translator PyCharm. Pycharm is a Python IDE that has two editions: Professional & Community. The version used for this project was the Community one that is free to everyone to use.

The following was used for the test methodology used during development:

[2] Can be found at https://networkx.github.io/documentation/stable/index.html

On the stage of language grammar definition and parser construction, the source code of yacc and lex parser was tested against a list of intended inputs the team agreed upon conception of the project. Then, each function defined in Honey.py was tested for expected outputs against a .csv file.

To verify that the translator is working according to our language definitions we created the following scenario.

Imagine you are a school principal and you have several students who are getting sick. You want to know how they got infected so you create a list that has the following:

16 lines (16 sloc) | 403 Bytes

Raw | Blame | History

Search this file...

	name	age	gender	grade	vaccinated	infected
2	Jessie	11	Female	5	No	Yes
3	Cesar	10	Male	5	No	Yes
4	Pepito	10	Male	5	No	Yes
5	Cody	11	Male	5	Yes	No
6	Karla	10	Female	5	No	Yes
7	Hannah	11	Female	5	No	No
8	Simmons	11	Male	5	No	Yes
9	Wilfredo	9	Male	4	No	No
10	Belinda	6	Female	1	No	Yes
11	Chris	8	Male	2	Yes	No
12	Paula	10	Female	5	Yes	No
13	Enrique	9	Male	4	Yes	No
14	Kirby	9	Male	3	Yes	No
15	Fox	8	Male	3	No	Yes
16	Zelda	9	Female	3	No	Yes

You start to analyze it. However, by the second day of work, you are notified that there are more sick students. Instead of manually copying the data, BEAR can be used to extract that information, create a graph separately for each list and unite them together.

That way you can know who got infected on the first day, who on the second and a graph that shows all the students who are sick.

Using BEAR you could add more graphs related to the case but that have extra attributes such as GPA of the student, date of birth, and so on.

For a live demonstration of how it was tested, please proceed to the live demo linked in the first pages of this document.

Conclusions

Final thoughts

BEAR is a programming language that thrives to make Social Network Analysis easier for those that don't have much programming experience. We followed the KISS (Keep It Simple, Stupid)[3] design philosophy.

We created the base in which this language program can expand into something more.

However, this is not the end. The team aspires to continue this project. The following are some of the future worked planned for this programming language:

- ➔ Introduce the notions of reading data streams from social network services such as Twitter
- ➔ Broaden analysis tools included in the language
- ➔ Introduce the notion of user-defined scripts
- ➔ Develop a more user-friendly interface
- ➔ Fix bugs
- ➔ Implement matPlotLib

The project at this stage marks a working start of a tool that has the potential of deciphering many mysteries yet to be discovered in the ubiquitous social networks that exist today and in the near future.

[3] https://www.interaction-design.org/literature/article/kiss-keep-it-simple-stupid-a-design-principle