Data Visualisation

Guichi Zhao

COMP5703
Information Technologies Project
Semester 1, 2015

Supervisor
Josiah Poon

07/06/2015

Data Visualisation Project

Abstract

The purpose of the project is about data visualisation.More specifically,providing a plain text file ,the program will draw a graph base on the it.Other than that,users can view specific aspect of the dataset in a interactive manner.Basically ,the program will help user view the dataset in a more clear and intuitive way as well as reveal some hidden information which is not obvious in a text file.

Introduction

The background of the project is Traditional Chinese Medicine (TCM).The raw dataset is a collection of prescriptions which consist of a set of herbs.The researcher analyse the dataset by reducing the size of original dataset to a equivalent one with the help of boolean algebra ,until the most significant herbs remain.Due to the existing reduction algorithm is NP-hard and the size of a particular disease dataset is quite large,usually hundreds of prescriptions and hundreds of possible herbs for each one.A NP-hard algorithm is not feasible, and researchers adapt it to a heuristic version and it works well.So it is important to get a intuitive sense of the dataset.
The dataset looks like this:

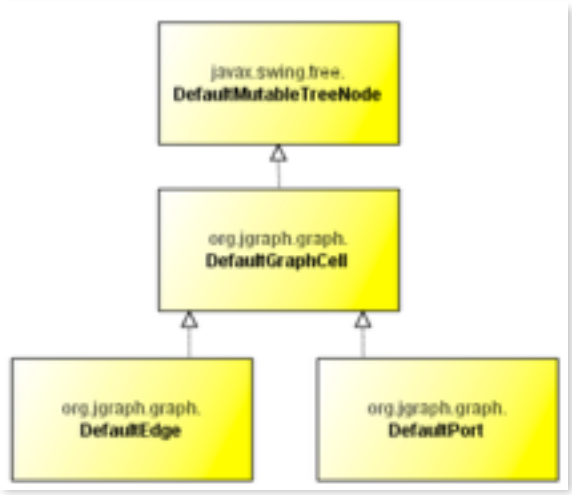
```
~VAR2~VAR4~VAR6~VAR9~VAR15~VAR17~VAR19~VAR21~VAR28~VAR3
~VAR3~VAR13~VAR15~VAR19~VAR21~VAR28~VAR90~VAR92VAR118~V
~VAR8VAR181
VAR9VAR239
~VAR13~VAR15~VAR17~VAR19~VAR21~VAR89~VAR90VAR124~VAR176
VAR17VAR113
```

Each line is a implicant which is roughly equivalent to a prescription and the implicant is composed of both positive and negative variables which is exactly equivalent to a herb.
Positive means the herb present in prescription while negative means not present. Only positive variable should be visualise and it is reasonable to ignore the negative ones.Because the negative ones makes more computational sense rather than a real life sense.

System Design

Background Tool

I choose Java as my implementation language and mainly two toolkit are used
The first one is native Java GUI API , Swing
The other is JGraph,although being a third-party library,it complies all the Swing standards and highly consistent with it.JGraph itself is an extension of JComponent ,which is Swing base class for all components .Also ,JGraph is similar to JTree(Swing API) in a number of ways.Some component inherit directly from



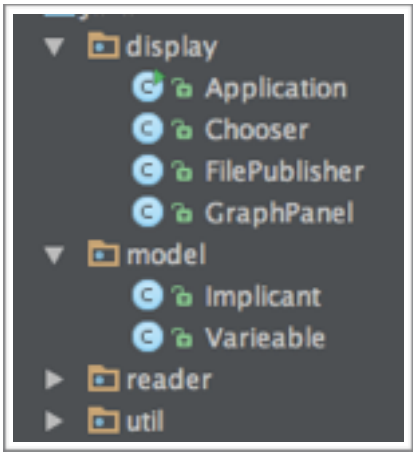
JTree component

System Architecture

The architecture of JGraph is Model View Controller (MVC), like Swing JTree and JTable.Also add more View layer features compared to native Swing library.

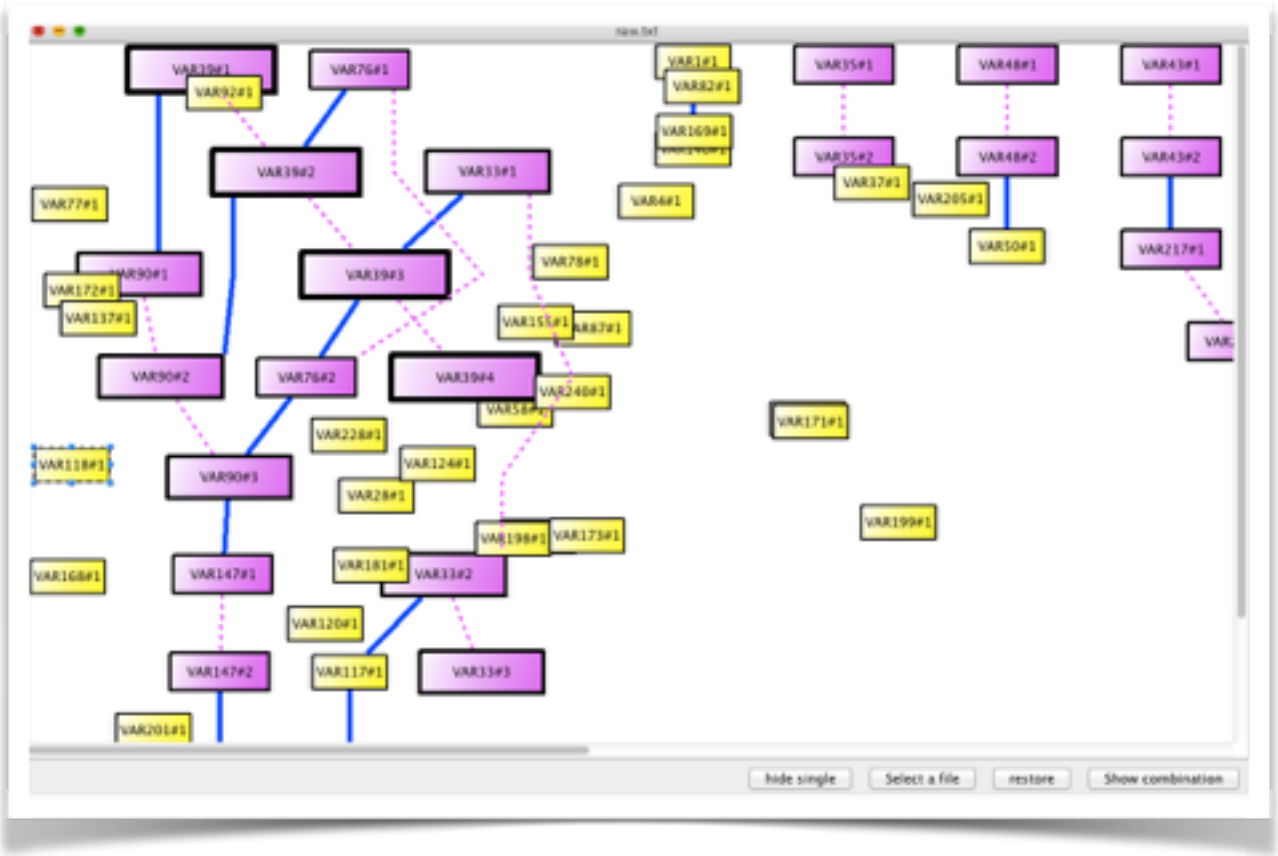
```
GraphModel model = new DefaultGraphModel();
GraphLayoutCache view = new GraphLayoutCache(model,
                                new DefaultCellViewFactory());
JGraph graph = new JGraph(model, view);
```

Model hold data about the graph and provides various methods to access data
View is extra layer logically above the model that perform the task of of visually presenting the graph
Controller is a bridge between model and view



As can be seen ,the structure of the project also conform to the MVC architecture,***model*** package include Implicant and Variable class which encapsulate some field and method respectively,***display*** package account for GUI component,and other packages are for some helper or utility purpose .

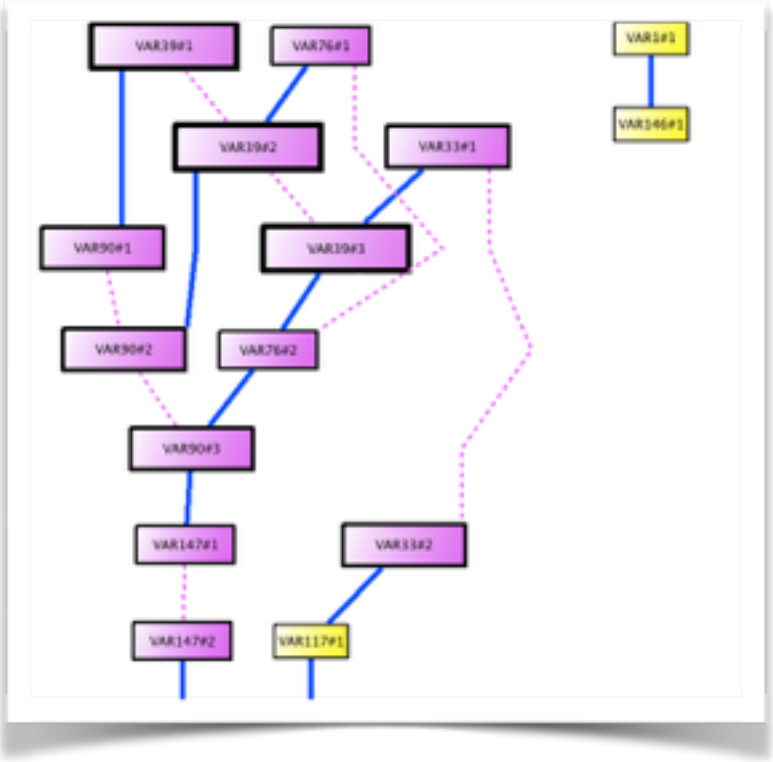
User Operations



When the user start the program:
The snapshot above appear

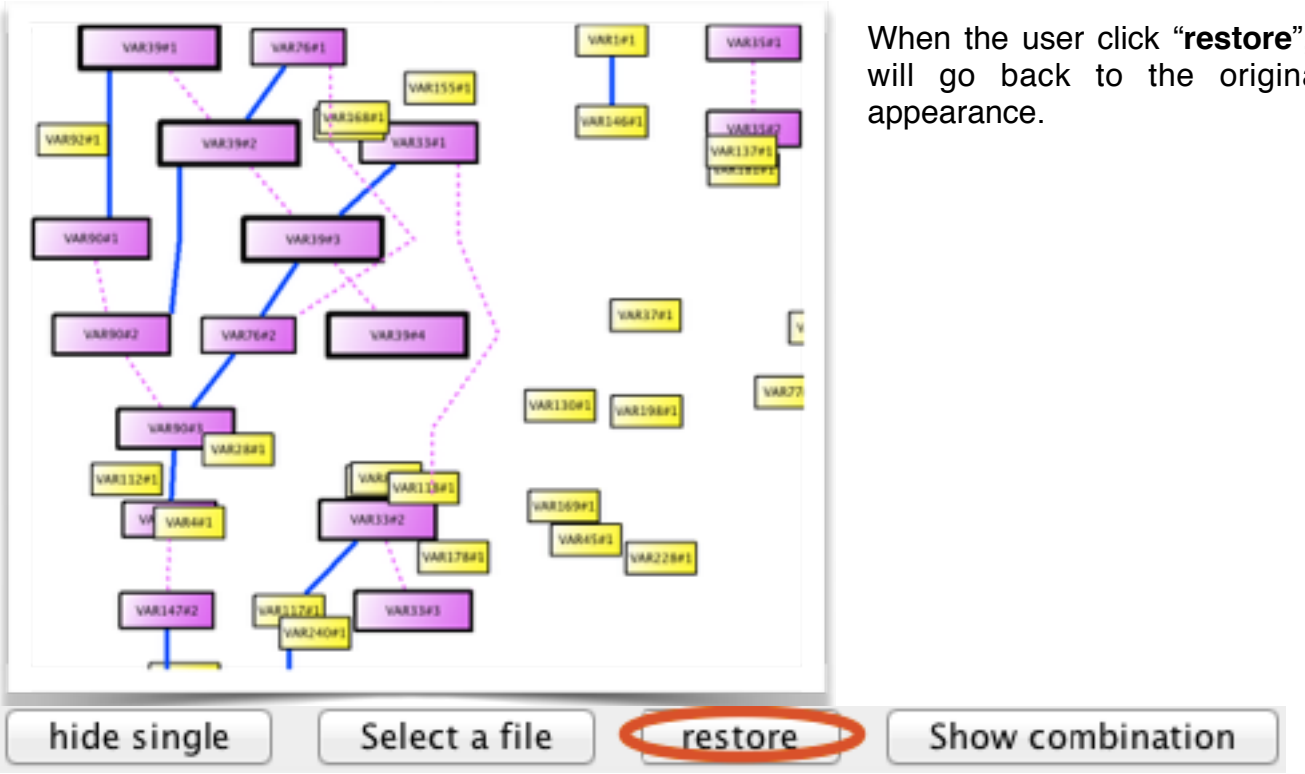
- If the variable present in ***only one*** implicant ,the colour is yellow .Otherwise,it will to be purple.Observing the graph more closely ,you will find the colour of purple cells is not exactly the same .Actually,the colour strength is based on the times the variable participant in implicants over the whole dataset.For instance,the colour of variable present in three implicant tend to be darker than variables presented in two implicants,through both are purple.
- Other than colour,cells are also differentiated by size and the thickness of border. The more time the variable present in different implicant ,the size is larger and the border is thicker.
- The cells are differentiated based on how many times appeared in different implicants because the cells appeared in a lot of implicants can be regarded as significant ones.
- Variables in same implicant are linked by solid blue line.
- If the variable present multiple time over the dataset,the same variable belong to different implicant are linked by dashed pink line.And the variable name is suffix by a hash tag (#) followed by a counter indicating the time it present.

As show by the graph above:
[VAR39 VAR90] are in same implicant.Similarly,[VAR76 VAR39 VAR90] [VAR33 VAR39 VAR76 VAR90 VAR147] are also individual implicants.And VAR39 seems a important herb on reason that it appears in multiple implicant.Such significance can be indicated by its large size and thick border.

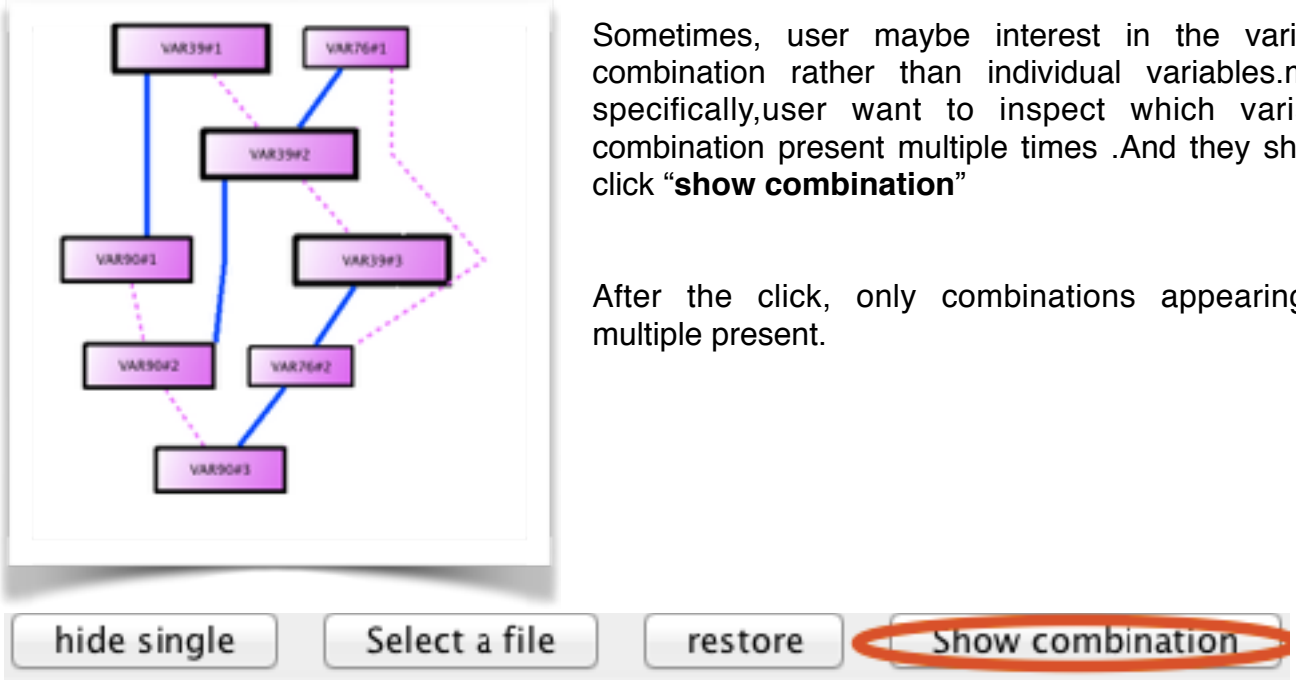


If the users think if a variable only appear once over the dataset is not of interest,he or she can click **"hide single**" button,which make the graph clearer.

Compare the original graph ,A plenty of yellow cells disappear ,because they are the single variable contribute to a implicant.Observing more closely,VAR39#4 also do not present because VAR39 also the only variable in a particular implicant although it present multiple times over the dataset.

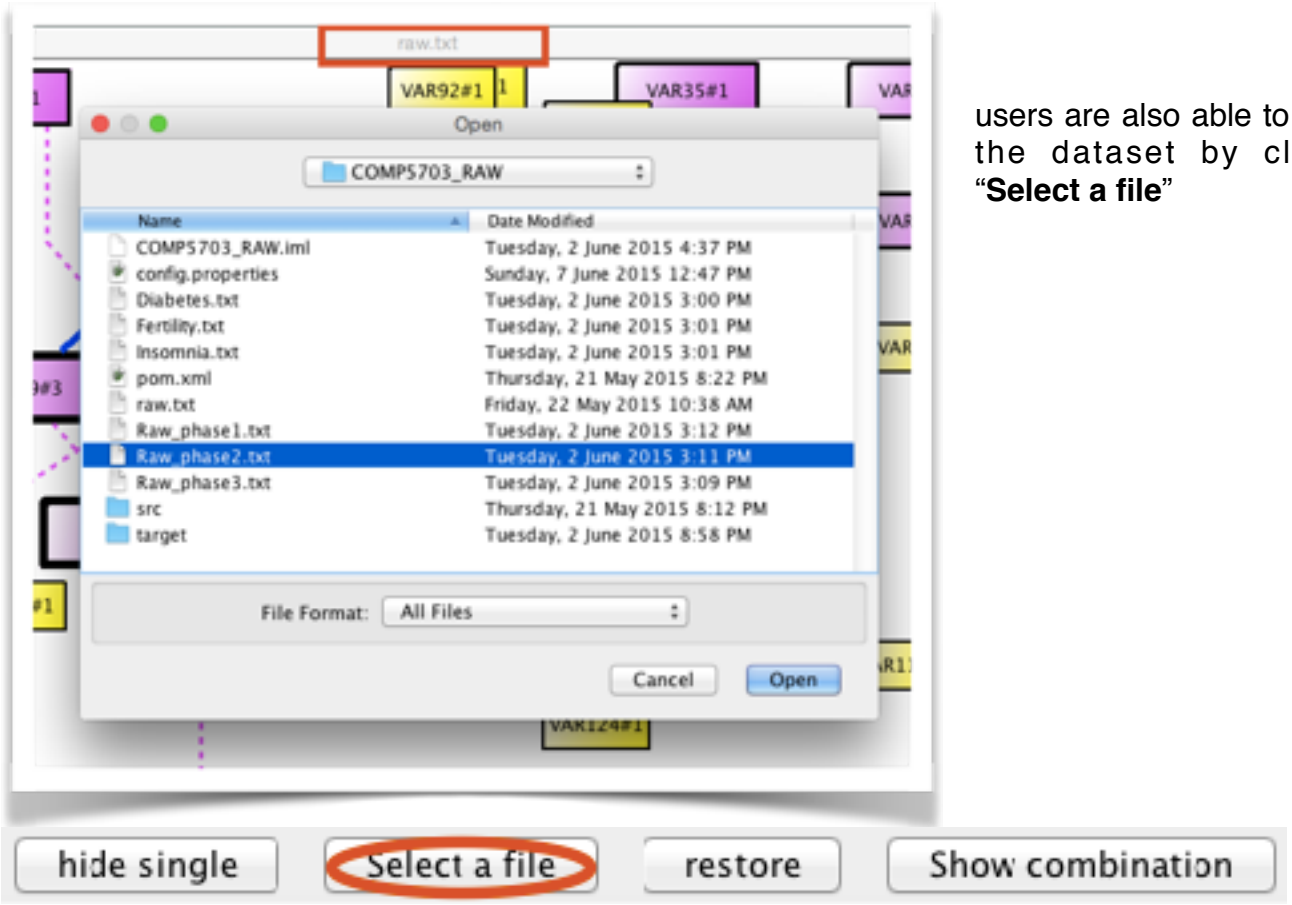


When the user click “**restore**”,it will go back to the original appearance.

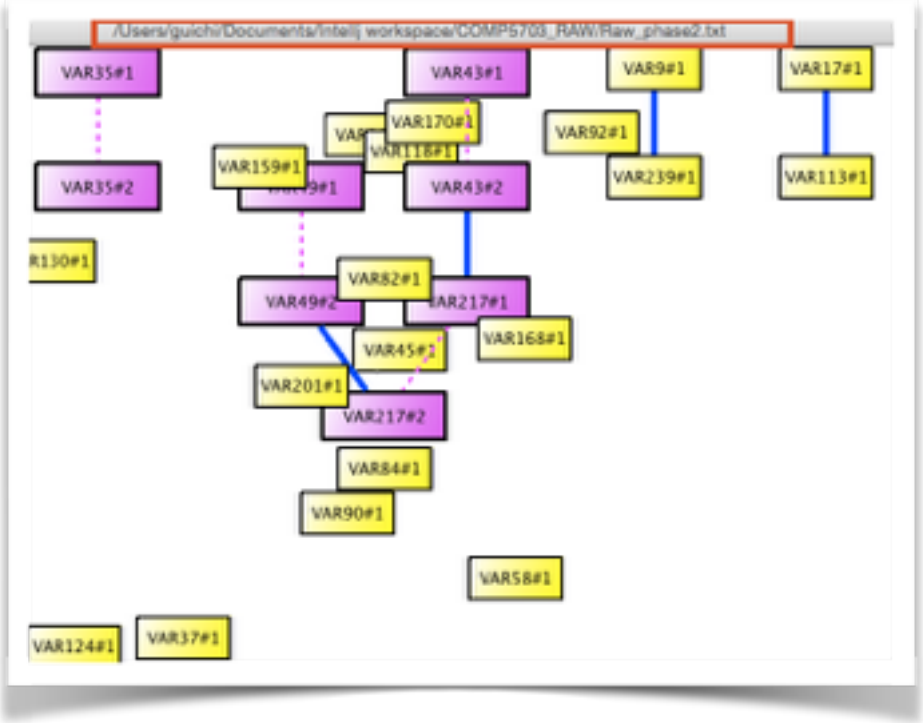


Sometimes, user maybe interest in the variable combination rather than individual variables.more specifically,user want to inspect which variable combination present multiple times .And they should click “**show combination**”

After the click, only combinations appearing in multiple present.



users are also able to switch the dataset by clicking "**Select a file**"



After the clicking ,a new graph will present base on the new datasource

Note that the name of the new file display at the top ,as highlighted by the red rectangular

Evaluation

problem

I evaluated the program by providing different input file and find that if the dataset goes large ,the program fail to works perfect

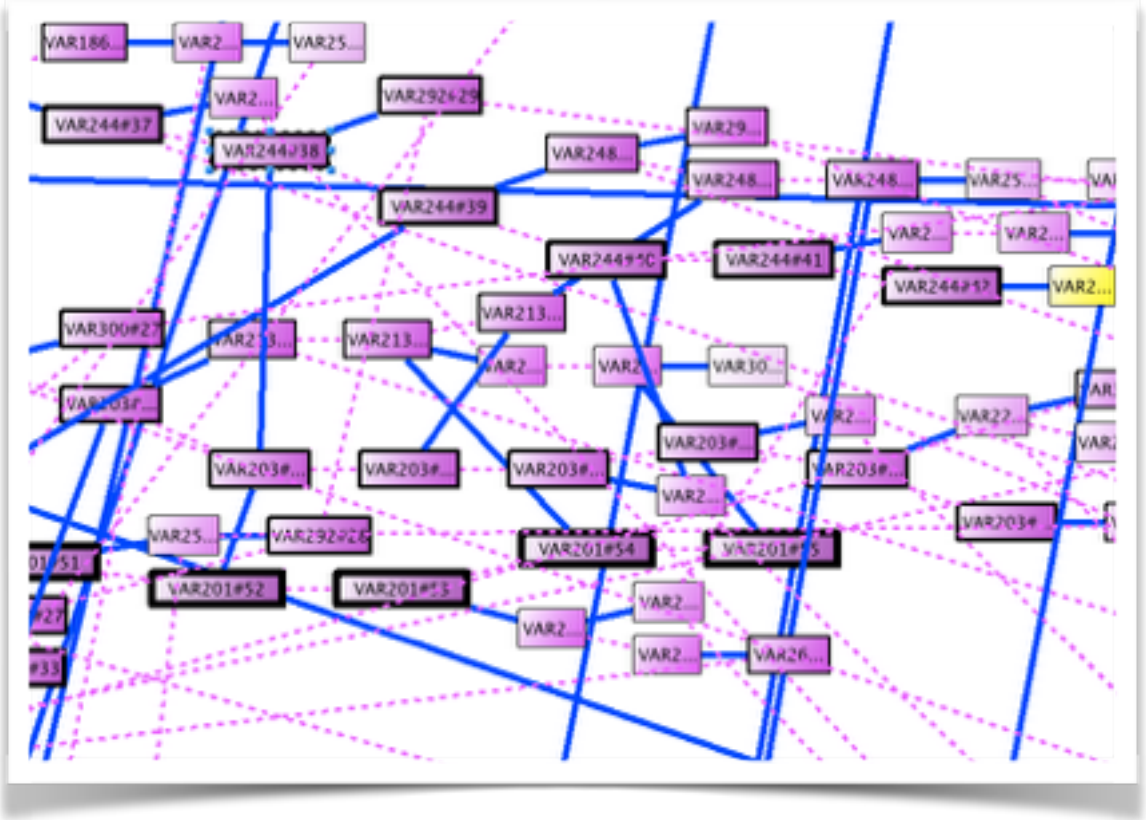
The maximum number of a variable appeared over the input is 4 times in original file,and the number goes to 55 for some dataset provided afterwards.The problem is that the layout algorithm,say,JGraphHierarchicalLayout do not apply

```
java.lang.IllegalArgumentException: Comparison method violates its general contract!
```

A possible solution is to change the layout algorithm to JGraphCompactTreeLayout

```
# decide if link between same variables from different present
showLinkBetweenImplicants=true
# 1->JGraphHierarchicalLayout  Seem to be the best option if do not display Link
# 2->JGraphCompactTreeLayout  will work if display Link
layout=2
```

and the graph like this:

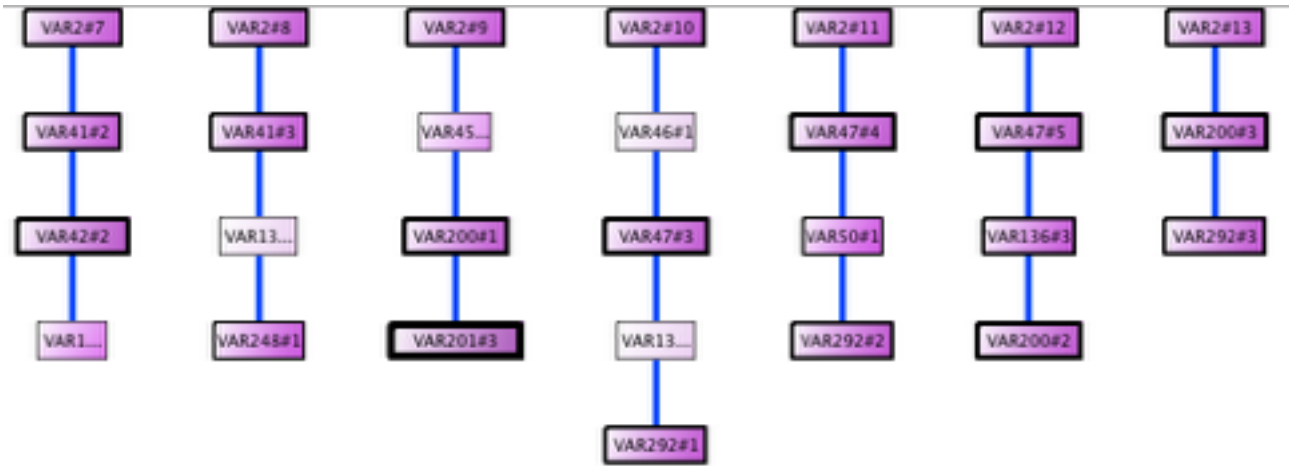


The display is quite messy,and actually not very useful.

Another option is to still apply JGraphHierarchicalLayout but compromise the link between the same variable from different implicant:

```
# decide if link between same variables from different present
showLinkBetweenImplicants=false
# 1->JGraphHierarchicalLayout  Seem to be the best option if do not display Link
# 2->JGraphCompactTreeLayout  will work if display Link
layout=1
```

In this way all the implicant display sequentially:



The information about individual variable preserved but lost information about the relationship ,which is not desirable.

Future improvement

I come up with two ways to solve such problem:
Firstly ,I can adapt the JGraphHierarchicalLayout algorithm from JGraph library to make it suitable.
Other than that,The way to indicate relationship between implicants can be changed.For instance ,not by dashed pink line linking same variable,rather,by a circle group.And due to the fact that the original way to display works well ,the display strategy can be switch base on size of data input.

Process

project plan and execution

Week 2&3&4

- Read the paper to identify what kind of dataset should be visualised
- Reviewed Java Swing
- Study JGraph user manual

Week 5&6&7

- Implement the project
- In week 7,supervisor told me I should visualise the original dataset rather than the final dataset,because the highly reduced final dataset lose most of the information.
- Also ,I was told that the users should be able to interact with the program

The information in final dataset is quite limited. And should not be the one to visualise.

Week 8&9&10
- Reimplement the project base on the dataset before processed.
- Make the program interactive base on supervisor's advices.

Week 11
- Demo the project
- After that.Supervisor pointed me some extra file ,which cause the problem

Week 12
- Tried to solve the problem

Tools and Skill

I chose JGraph as my third-party library.I find it basically a bad idea in about week 7,the community stop maintaining and updating it since 2010 ,so when I encounter a problem ,it hard to get much help online.Actually,the JGraph community is quite active now,but it is migrate to mxGraph which is a javascript library although share the same design and principles with Java implementation.

Reflection

Difficulties

I had encountered problem such as "when the user upload a new dataset,how the graph can be updated accordingly" I searched online,and someone mentioned a possible solution with Observer design pattern ,I recalled I learnt such pattern before ,but not so impressive.I picked up the textbook and studied the corresponding chapter.Finally ,I applied the Observer pattern to the project and perform the functionality.

Lessons learned

- Chose a active tool to implement the project.Especially for a individual project,joining a community and share problems and ideas is a good option.
- Contact to supervisor more frequently .Sometimes ,I suppose I had completed the task and it is OK.However,there **are** some problems.Actually ,after a meeting with the supervisor,improvements can always be pointed.The most significant lesson is that,when I receive the extra dataset,it even can not work as expect.Should I got the dataset earlier,maybe I can get sufficient time to handle it decently.

Conclusion

To conclude,the program is able to visualise each implicant and the roles of individual variable in different implicant for relatively small dataset.However,for a larger dataset,the linkage between different implicants should be compromised.In future work,the layout algorithm can be modified to adapt to the current case,or a new way to visualise the dataset can be introduced.

Appendix

source code

https://github.com/GuichiZhao/COMP5703_RAW