cryptoGraph – Software Design Document

# Introduction

*Briefly discuss the functionality of the program.*

In my exploration of the available data, I found that:

* exchangeInfo.json contains useful information about potentail trades. Each trade between two assets is called a symbol. This file contains information as to whether trading is allowed, and contains field that specify the baseAsset and quoteAsset.
* But exchange ifor does not contain the prices of actual trades, or the volumes. We would like to know this information for our weighted edge graph
* the 24hr.json file

# Installation

*Requirements, dependencies, description of files/directories.*

# Terminology/Abbreviations

# Walkthrough

*Demonstrate all functionality, indiating anything that isnt’t implemented or working properly.*

# Future Work

*Missing items or suggested enhancements.*

Linked lists have been used extensively in this implementation, but more thought will be given in this section to implementation of different data structures.

* Edge attributes stored as a dictionary type structure using a hash table?
* When reading off top 5s/ top 10s, store in a Queue

### Asset Filter Section:

The asset filter operates on the graph, and removes vertices and edges containing the chosen assets. However, this does not affect the results from ***Option 3: Find and Display an Asset***. This is due to the fact that Option 3 is based on reporting from the BinanceData object, which is a container for the json files.

The **Asset Filter** updates the graph, and effectively removes vertices from paths when doing graph traversals. But is does not affect the resutls from ***Option 3*** or from ***Option 4: Find and display trade details***.

In this way, the reporting is really broken up into two categories, those that report on the raw data, and those that report on the data contained in the graph. As a suggestion for the future, I think that this should be unified, so that asset exclusions exclude assets from all reporting.

### Classes and Inheritance

Better to combine DSALinkedList and DSALinkedListDE. This would only mean modifying the DSALinkedList.removeValue() method to deal with tail attributes that are additionally present in the DSALinkedListDE class.

# UML Class Diagram

# Class Descriptions

*A description of any classes you have, you need to let us know not only what the purpose of that class is but why you chose to create it. As part of this, also identify and justify any place where it was possibly useful to create a new class ut you chose not to, expecially when it comes to inheritance.*

# Optimisations

Durng testing, an issue was identified when calculating the cost of trade paths. Using the data provided for the task, the only apparent trade price information was contained in the 24hrs.json file, that summarises all trades in the last 24 hours. This data was used to extract edge weights for the graph, so that the edge weight was equal to the average trade price for all trades of that symbol in the last 24 hours.

However, some trades are uncommon, and while they are permissable, may not have occurred in the last 24 hours. To this end, rare trades would be assigned a cost of 0.0.

This was problematic when calcualting the cost of trade paths, as that combinent of the path would be assigned a weighting of 0.0, thereby making the overal trade cost = 0.0.

I attempted three solutions to this problem:

1. leverage the Binance API to get currnet trade price for each symbol individually

https://api.binance.com/api/v3/ticker/price?symbol=ETHBTC

This proved to be very slow in practise.

1. leverage the Binance API to get currnet trade price for each symbol individually, but only for those symbols with no trades in the last 24 hours

This also proved to be very slow:

if cost == 0.0:  
 symbol = fromLabel + i.\_label  
 cost = self.getSymbolPrice(symbol)

1. leverage the Binance API to get current trade prices for all symbols in one get request, and use this to update all edges in the graph iteratively in Python

# References

*In Chicago style*.

DSA Assignment Notes

1. Read in the files
2. Make sure you can **see all the assets**
3. she got it down to about 10 or 11 assets
4. Which field are you going to use? Could just use a subset, and then exnad to the others later.
   1. Pick a few like volume count and average and implement with that
5. She wants us to extract the data, have it as a graph, and get some of the information out
6. Build a graph of the trade
7. Must be directed
8. Not everything will be conencted to everything else
9. Must be a direct path through other assets to define the trade
10. Start with just getting a graph
11. Will need to implement edges

this is all about the edges

1. List the top 10 most connected, the ones with the highest volume, the msot count, the most weighted price something like that
   1. so just a few top 10s for report mode is enough
   2. put in the ‘most connected’ ones
2. We can use python tuples (but not lists)
3. Put in a comment to day that you realise you are working with a dictionary if you are…
4. Hand translate the json file rather than understand how to use a json reader and convert that dictionary back out… if you want. Otherwise there is that video link on a json reader…
   1. but yeah, we can use a json reader package I think she said
5. This is a good real world thing
6. Get it so it RUNS – maybe do this as a github repo so that if I make changes and break it, at least I can go back to a working version
7. The report is worth a lot, if I cant implement the last bit of functionality at least write up a good report
8. We can use Gephi or another package as a vaidation step if we want – but not necessary this was just one persons idea
9. **I have used some of my functions and classes from other practicals - how do I cite them?** You need to add a comment in the block documentation at the start of reused classes or methods, as well as noting the code was previously submitted for Prac X in the justification documentation.
10. **Do we need to include test harnesses for data structures and ADTs implemented in the practical exercises?**Yes
11. **To what extent can we research on ways of implementing certain aspects of the assignment?** You need to cite any algorithms or code you use from other sources. I don't expect you to reinvent or come up with your own tree varieties. **However**, there should be no verbatim code (identical) used in your assignment (except that used and cited from your own practicals). **You will not get marks for any copied code - even if you cite it.**
12. **Can I use list/ArrayList or other built-in datatypes?** - No, you shouldn't use any of the Java/Python built-in ADTs, but you can implement versions of them. Again, you will have to justify any choices made. Use of Java/Python ADTs will incur a large penalty (previously -20 marks) - as implementing data structures is the point of this unit. *In Python, using a list output by split(), and readlines() is an exception.*  
    If you are choosing to use a visualisation package - if it needs specific data structures/types, you can use them for that specific part of the assignment.
13. **Can we refer to the lecture notes when writing our documentation (e.g. for finding some of the advantages and disadvantages of using one data structure over another)?** Yes.
14. **Will we lose marks if some of the algorithms we use differ from the ones used in lecture slides (ie if we create our own version of sorting algorithm)?** That's fine, but if you've found it on another source, you should reference where you found it.
15. **Can I use the supplied test harnesses?** Yes, but you need to cite them, as with your other code from other sources.
16. **Can I use JUnit or Mockito?** Yes, use of testing frameworks is encouraged. Bonus experience, but no bonus marks...

**MARKING**

* CODE demonstration is only marked on whether it passes tests! **30 marks**
* Implementation **30 marks** for code quality and unit test implementation.
* **Must submit declaration of originality/cover sheet document**

**Implementation steps to do:**

1. get the json data and read it in to python (into what? each asset is a vertex, each edge is a trade or trading price.) ***is it a trading price, or heaps of edges between each asset each one is a historical trade?***
2. make an API request for a file with 10 or 11 well connected assets. – did she share this?
3. which fields are we going to use? just use a few at first.
   1. volume
   2. count
   3. average price
4. Build a graph of the trades (of the possible trades I would sa… not past trades)
   1. directed
   2. ETHBTC = ETH->BTC in that direction
   3. you can follow a path through the trades
   4. this is all about the edges
5. Stats
   1. the top 10 highest volume
   2. the top 10 most connected
   3. the top 10 with the highest count
   4. the top 10 with the highest weighted price

***if anything is weird, then document my interpretation of what they mean –***

***‘I assume volume to represent the …’***

* 1. overall count for number of assets, number of connections

***check in ‘trade overview’***

The data

exhangeInfo.json just shows what currencies are tradeable… we want to know the latest trade price I think… otherwise what do we do for the edges?

# WE WANT TO USE THE 24HR json!!!

Has one object for each trading pair and its average price over the last 24hrs.

**Good idea here:**

You mentioned in the lecture that we have to implement a directed graph..

As far as i understood, the ask price is the lowest price that somebody would sell ETH for BTC (1 ETH for 0.03308000 BTC), and the bid price is the highest price somebody would pay in BTC for ETH (0.03307900 BTC for 1 ETH).

Therefore, if you pay 0.03308000 BTC you will get 1 ETH (BTC --> ETH) and if you pay 1 ETH you would get 0.03307900 BTC (ETH --> BTC).

Wouldn't that be an undirected graph?

All coins for MY account:

<https://api.binance.com/sapi/v1/capital/config/getall>

**Get individual asset names?**

They seem to be only list as ***base assets*** of a trading symbol. Use this to get a list of them.

**OK, so we have the symbol and its baseAsset and quoteAsset in one file**

**And the weight of that edge in another file.**

**Create the edges from one, without weights**

**Then go through anp apply the weights using the other file!**

**Where are we at here? After big break for Databases Unit**

Cryptograph runs.

Test harness runs.

1. Load Data – runs
2. Find and display an asset
   1. runs
   2. shows ‘Possible trades’ only
   3. What other details should we show?
   4. what else to show? There is some talk on Piazza About this.
3. Find and displayt Trade Details
   1. says ‘No trades in the last 24 hours if the symbol is not foung, but I think all valid trades are in that file, so it should check if it’s a valid trade first… A method to check if it’s a valid trade?
   2. Leave for now, OK
4. Potential Trade Paths
   1. count edge wieghts either as we go, or calculate after finding path
   2. Calculate commission at the end.
   3. *this will be documented online, find best path , weighted edge graphs*
5. Set Asset Filter
   1. works well, note in documentatoin that this only applies to the trade paths, and that all trade details will still be visible
6. Asset Overview
   1. **not sure what to do**
   2. ‘As statistics, or some other representation’
7. Trade Overview
   1. E.G. Top 10 fro price, volume, and count
   2. (maybe make that top 10 for lastPrice
8. Save data (serialised)
   1. probably should serialise the Graph object, and *maybe*  the Binance Trading Data object.
9. Exit
   1. print a message maybe
10. **Extentsion**
    1. do last, but I think that we could provide a ‘get latest data from Binance’ option that uses the API

Don’t know much about the bitcoin trading terms

Quote coin – there is Base coin (the first one in a symbol) and quote coin the second one. Trades go from base to quote. Consider BTCETH – a trader could have 10 BTC (the base asset) and want to trade all this to ETH. This trade would have a base volume of 10.

Quote Volume – when you say how many of the quote asset you want to buy. So instead, say the trader wanted to buy 3 ETH coins, they could specify a trade of BTCETH with a quotevolume of 3.

It appears that volume \* weightedAvgPrice = quoteVolume

So volum must be quivalent to saying baseVolume

**Must include commission** – but we can ignore discounts using binance coin.

Its 0.1% on all trades I think.

“If you are not using BNB (Binance Coin) to pay your trading fees, each trade will carry a standard fee of 0.1%.” from <https://www.binance.com/au/support/articles/115000429332>

**Idea on implementing edge weight path search**

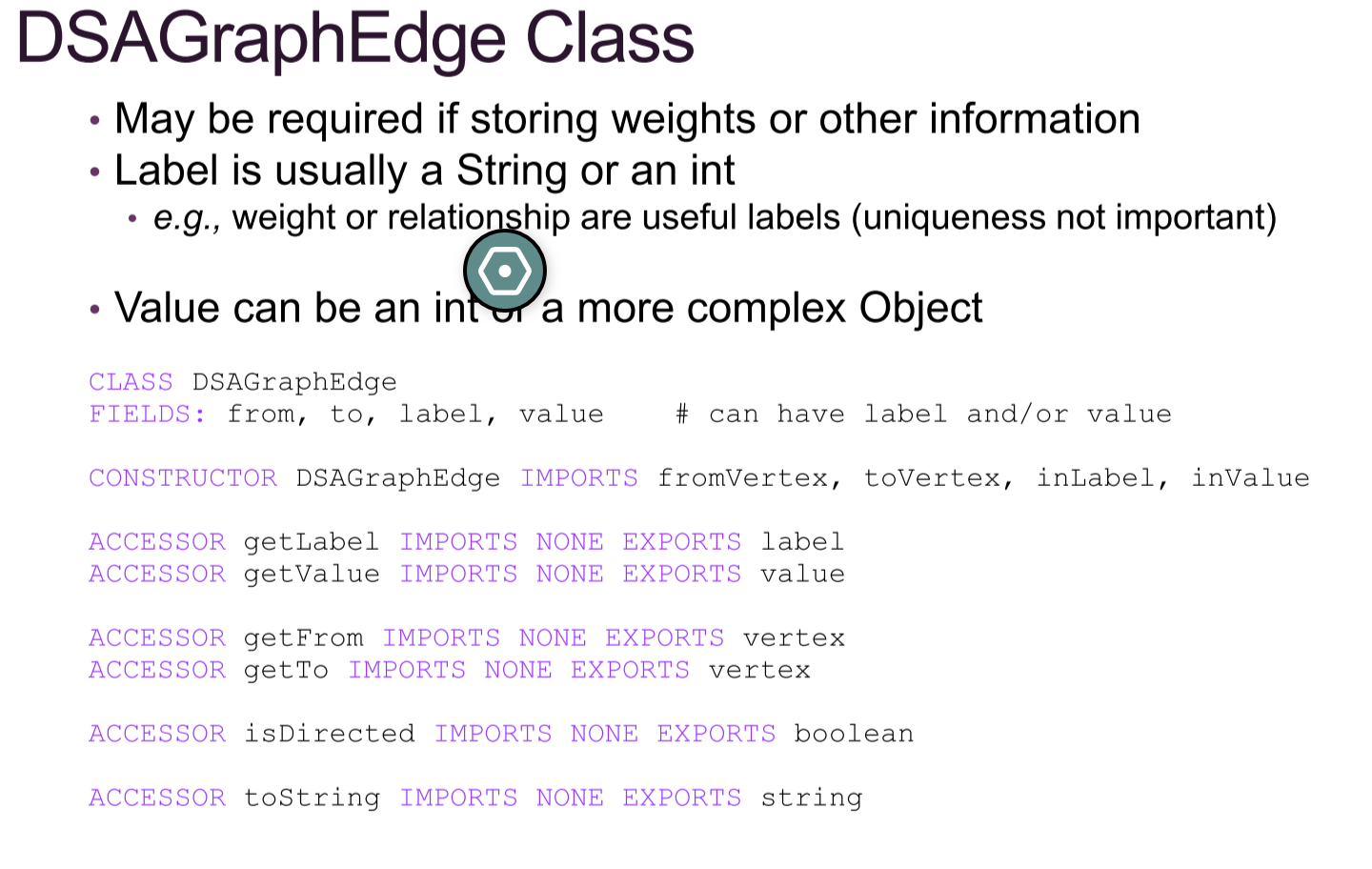
1. get paths normal way by mapping ther vertices
2. then given a path go through and find the sum of all the edge weights
3. then add the commission

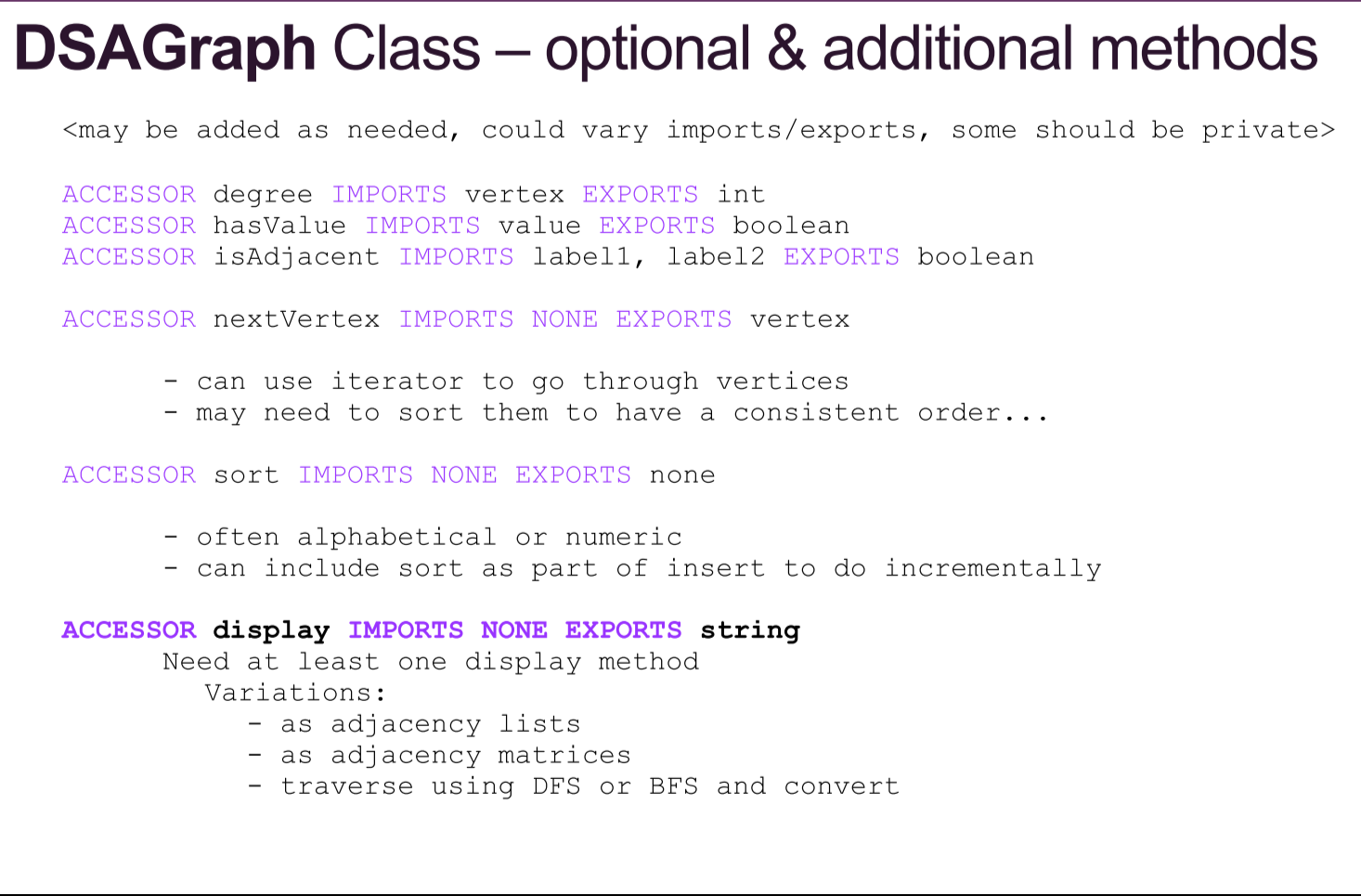
# Action Plan

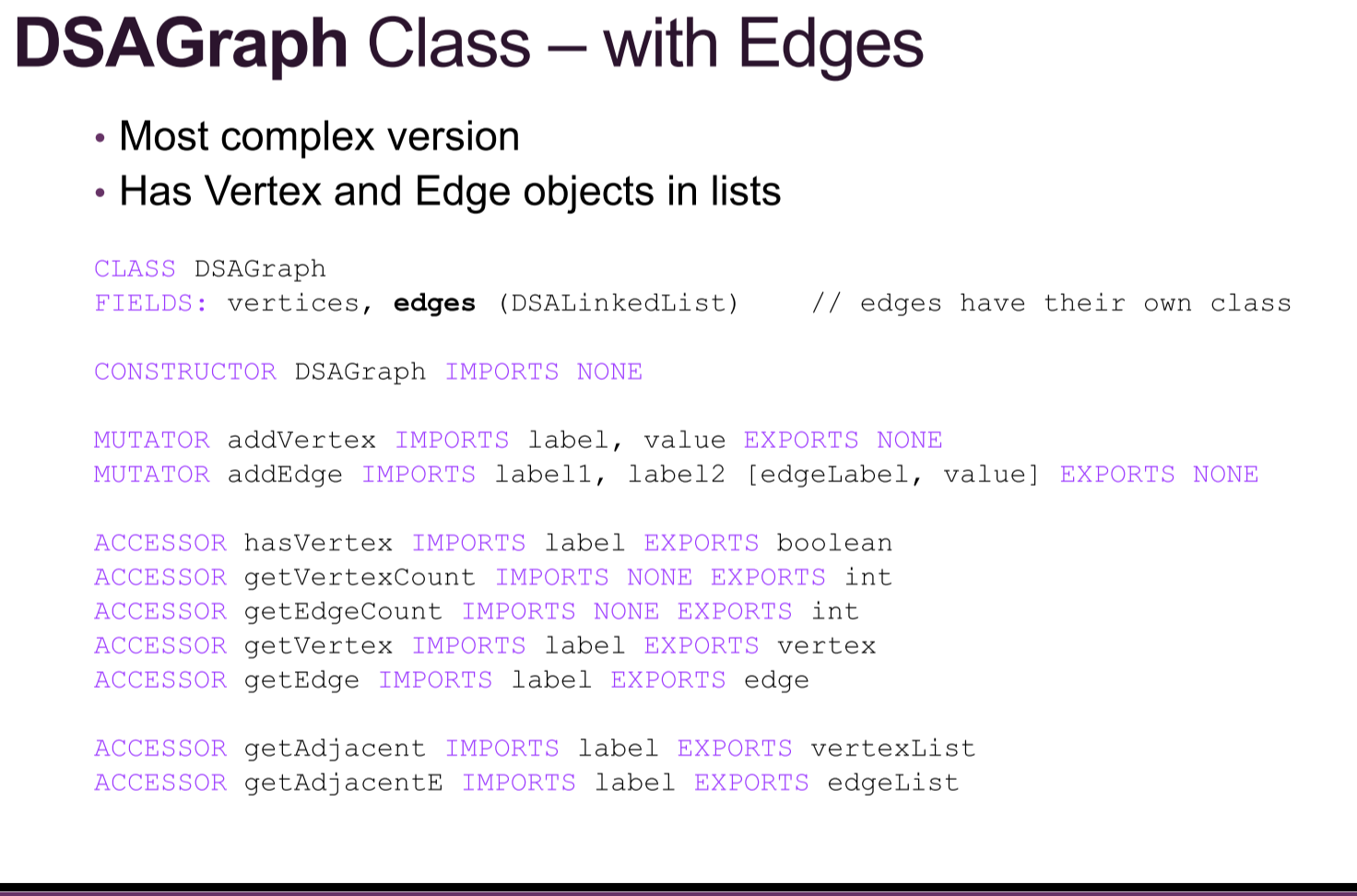
**I have the overall architecture down**

**Now just refactor.**

1. DFS or BFS, make list of edges though.
2. Base case is if edge.toVertex = Destination of path
3. USE THIS: The graphs lecture including code for DFS/BFS, I was assuming that was how you were traversing your graph to find paths.







**Note son API integration**

REST is a set of conventions for stucturing Web APIs

* make a request to a URL
* get relevent data back
* GET and POST methods most common
* POST creates something on the server I think, get just gets something

# NEXT TODO

Merge sort method for the Paths Container

Should the json files be specified as commandline arguments?