Contents

[1. Introduction 2](#_Toc20821096)

[2. Graph Visualization Metaphor-Flow Map 2](#_Toc20821097)

[3. Framework of the design 3](#_Toc20821098)

[4. Specification of the design of visualization 3](#_Toc20821099)

[4.1 Layout design specification 3](#_Toc20821100)

[4.1.1 Map 3](#_Toc20821101)

[4.1.2 Nodes 4](#_Toc20821102)

[4.1.3 Edges 4](#_Toc20821103)

[4.1.4 Readability 4](#_Toc20821104)

[4.1.4.1 Icons 4](#_Toc20821105)

[4.1.4.2 Colors 4](#_Toc20821106)

[4.1.5 Labelling techniques 5](#_Toc20821107)

[4.2 Graphics design specification 5](#_Toc20821108)

[4.3 Attribute Mapping 5](#_Toc20821109)

[5. Navigation 5](#_Toc20821110)

[5.1 Zooming 6](#_Toc20821111)

[5.2 Mini-Map 6](#_Toc20821112)

[5.3 Evaluation 6](#_Toc20821113)

[6. Conclusion 6](#_Toc20821114)

[References: 6](#_Toc20821115)

### Introduction

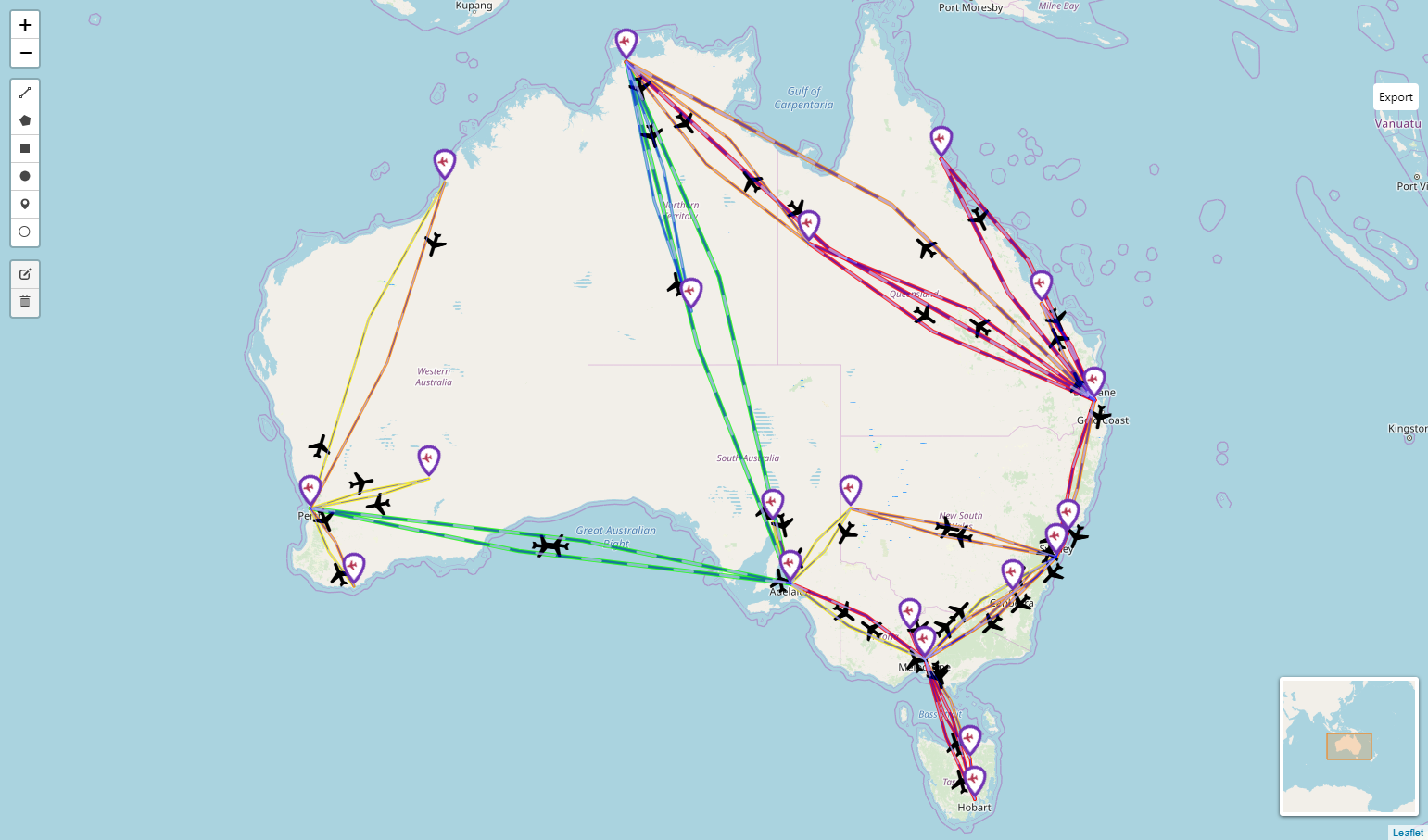
This report explains a design of graph visualization for flight data. The graph visualization is a “Flow Map”, which aims to show flight's <From -> To> relational structure as well as other information such “Air Space Class” and “Price”. The report explains how the designed visualization model achieve readability, usability with respect to Human Cognition Process. A prototype of this design can be checked through link: “<https://ouliang0128.github.io/DVVA_ASS2/>”.

The rest of this report is organized as following: Section 2 describe the chosen “Flow Map” and explain the reasons for the choice. Section 3 is the high-level description of the model, while the specifications of the layouts is discussed in section 4. Section 5 introduces navigation techniques for the graph visualization. Section 6 is the conclusion of this report.

### Graph Visualization Metaphor-Flow Map

In this design the graph visualization metaphor is “Flow Map”. As Doantam et al. (2005) clams, flow map have the ability to interpret movement of objects and reduces visual clutter. In consideration of the represented data is information about flights routes, flow map provides intuitive explanation about the data.

For example, we can see in **Fig. 1**, cities in the data file is represented as markers in the map. Combining with lines between those markers, users can have a perceptual intuition about the distance between the starting city and the destination city. This geospatial feature from the original data is hard to be recognized by human with other layouts, such as tree diagrams and scatter plots. Another advantage of flow map is that it represents the direction of each flight record by the direction of the plane icons alongside these lines (Fig. 1) and animations of these lines. With this implementation, users can easily find out where is the his/her interested flight in the map.



**Fig. 1**. Overall view of “Flow Map” design for flight data

### Framework of the design

For implementing a flow map on the fights data, this design utilize “folium” library of python, and realized it on google colab platform, the link of the design is as following: <https://colab.research.google.com/github/ouliang0128/DVVA_ASS2/blob/d3ae5ca0473e6ddfe261ad3fce31ed94457025b6/Flight_Visualiztion.ipynb#scrollTo=7W-IoQNChEei>

The main characteristics of this model are:

1. Extracting information of cities and draw them as markers in a geographical map
2. Drawing lines between two cities for showing flights information
3. Defining navigation scheme for displays detailed flight information, such as price, air space class and engine model of airplanes.

In archiving the above functions, 3 main steps are implemented:

1. Data pre-processing
   1. Find characteristics of each column
   2. Mapping city names to geographical latitudes and longitudes
2. Map Drawing
   1. Defining the parameters of nodes and edges
   2. Displaying all information on the flow map
3. Map Exporting
   1. Generating .html file for the flow map
   2. Upload the .html file to github page

### Specification of the design of visualization

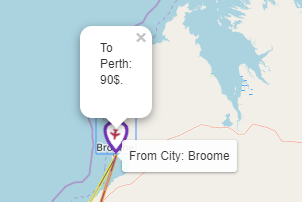
This section will explain the specific layout techniques includes for the design. In fact, all the graph drawing techniques are from “folium” library of python. Folium includes sophisticate interactive geographic maps as drawing background, in this design “Google Map” is selected as default. It also includes many tools for drawing objects on a graph. The flowing subsections will discuss those techniques included in this design

### Layout design specification

### Map

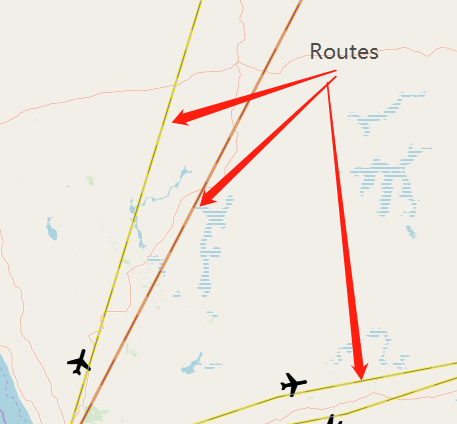
As it is discussed before, the background map is a copy of google map. Because the dataset only includes flight information in Australia, this design assigns the central latitude as -25.27, and central longitude as 133.77. In addition, initial “zoom\_start” is set to 4, so that only the proper neighbourhood area of Australia is displayed rather than the entire world map.

### Nodes

Nodes in the map represent cities in the dataset, they are implemented by “Marker” (**Fig. 2**) objects introduce by folium. For each node (city), its latitude and longitude are calculated before drawing. Due to the fact that there are repeated cities in the original dataset, the “unique” function from padas library is used for drawing one city instance once only. In addition, because there are only 20 cities included in the dataset, node-overlap problem is not serious, by setting “marker” to be the default size the potential node-overlap problem can be alleviated. Besides, the zooming technique that provided by google map is another way to solve the node-overlap problem.

**Fig. 2.** Node layout specification

### Edges

Edges in the graph represent the flight routes with is displayed by a combination of “AntPath” and “PolyLine” objects (**Fig. 4**). An AntPath object has the ability to show an animation of a movement between the starting point to the destination point, this feature help users to recognize the direction of the flight. The other object, PolyLine, has the ability to show color and width according to certain information of flight data. The implementation of PolyLines in this design also displays “plane” icons as their “text” attribute to help showing the direct of the representing flights. To avoid edge-crossing issue, the drawing of “AntPath” and “PolyLine” objects includes an intermediate point to separate two lines with opposite directions/

**Fig. 3.** Route layouts by using AntPath and PolyLine

### Readability

This design introduces several techniques to enhance the readability of the layout. These techniques guarantee the user accessibility of all information from the original dataset without overwhelming display it as a whole.

### Icons

City node are shown with spinning icons, which imply they are starting and ending points of flights.

### Colors

Color of a certain route implies the “Air Space Class” from the dataset, for which “A” is presented as “red”, “B” is presented as “orange”, “C” is presented as “yellow”, D is presented as “green” and “E” is presented as “blue” (**Fig. 5**)



**Fig. 5.** Color of routes identifies values of “Air Space Class”

### Labelling techniques

All the labelling techniques are implemented as navigation, which will be discuss in section 4.

### Graphics design specification

As it discussed in the previous section, the two major types of objects this design includes are edges and node. All attributes in the original data file are mapped into these objects.

### Attribute Mapping

There are 6 attributes in the original data file:

1. AirSpace\_Class:

This attribute will be displayed when the mouse is pointing at a certain route. Moreover, colors of routes are another way to show this attribute.

1. From\_City

This attribute is presented by starting points of AntPaths, city name can be seen by pointing the mouse to a certain node.

1. To\_City

This attribute is presented by ending points of AntPaths, city name can be seen by clicking the mouse to a certain node.

1. Price

This attribute will be displayed when the mouse is pointing at a certain route and clicking a certain node. Moreover, widths of routes is another way to show this attribute. The wider the route the higher the price of the flight.

1. Aircraft\_Model

This attribute will be displayed when the mouse is pointing at a certain route.

1. Engine-Model

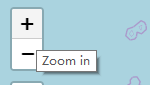
This attribute will be displayed when the mouse is pointing at a certain route.

By applying the above mapping, customer can easily identify cities and routes, other flight information can be revealed by interaction with nodes and routes. In consideration of the stability of the layout, because this layout uses google map as its background, adding new records of flights in other countries will not be an issue.

### Navigation

This section explains navigation techniques included in the design and evaluates them with Fitts's law.

### Zooming

Zooming is the main navigation technique in the design, with which users can inspect a local area in the map. To do this, one can use mouse wheel or click “Zoom in” and “Zoom out” buttons (**Fig. 6**) in the top left corner of the map.

**Fig. 6.** Zooming Buttons

### Mini-Map

When zooming in to a specific location, users may lose background information and feel hard to transform the view to another location. To solve this issue, Mini-Map is included **(Fig. 7**), which is displayed in the right-bottom corner of the map. With Mini-Map, users can have an overall view of the map, thus maintain the background information of the locality. In addition, changing focus on the Mini-Map is much faster than that on the “big map”.

**Fig. 7.** Mini-Map navigation

### Evaluation

Fitts’s law define the efficiency of graph navigation by :

*Selection time = a + b log2 (D / W + 1.0)*

Where D is the distance to the centre of the target, W is the width of the target. In this design, because widths of routes are wide, the navigation time is short. In this sense, interactive operation can be efficiency.

### Conclusion

This report explains the design of graph visualization for flight data. It describe how the “Flow Map” is implemented and all techniques included in design. Evaluation and justification of all techniques are elaborated in this report. The results shows that the implementation is basically successful, although some improvement can be done in the future. Possible improvements may include implementation of 3D layout and great circle routes.

### References:

Doantam, P., Ling, X., Yeh, R. & Hanrahan, P. 2005, 'Flow map layout', pp. 219-24.