FlightView: A Flight Information Visualization

Nicole Arksey

University of British Columbia

ABSTRACT

FlightView is an information visualization which displays several days of possible flight information for specified routing, designed for travel agents who book corporate travel. The visualization enables travel agents to quickly look through days worth of information to find the best flight for their customers.

The main visualization window displays nodes which represent the cost, times of flights and airlines. A Bifocal lens is implemented to help users reduce visual clutter and magnify areas on the visualization. More detailed information about the flight and possible saved flights are displayed at the bottom of the visualization. Dynamic filtering is also included to help users reduce the set of flights returned. FlightView was designed with the help of real users at BTI Canada, a corporate travel agency and has the potential to solve a real-world problem.

After the prototype was developed in Java using the prefuse toolkit, user feedback was gathered from 7 users at BTI Canada. Users generally like FlightView and had feedback on possible improvements. This information is compile possible further design changes. The next step for FlightView includes implementing some of these recommendations from users, investigating some outstanding usability issues and completing more formal evaluations. Initial reactions to FlightView show that it is a viable solution to the real-world problem.

CR Categories and Subject Descriptors: H.5.2 [Information interfaces and presentation] User Interfaces

Additional Keywords: Information Visualization, Bifocal Display, Focus + Context, Dynamic Filtering, Flight Information

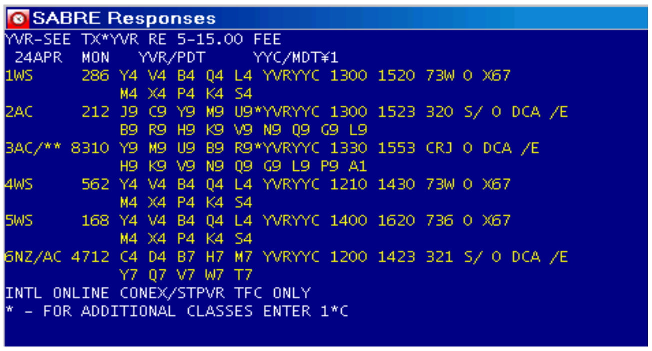
1 INTRODUCTION

Those who have tried to book a flight online might have experienced the frustration of trying to find the best flight to meet all of your needs. These might include trying to find the lowest fare, the shortest flight, a flight at a certain time of day and maybe with a specific airline. There are many different dimensions to choosing a flight and it is challenging and sometimes frustrating to sort through all your possible options. This gives us a glimpse of understanding the difficulty a travel agent experiences trying to find flights for different travelers many times in a day.

To gain a better understanding of these issues, I worked with BTI Canada, a corporate travel agency. BTI Canada has 6 different call centres across Canada and over 600 travel agents booking travel throughout the day for thousands of different customers. A travel agent at a call centre has extra pressure to find a flight that meets their customers needs within a certain amount of time to enable them to move onto the next client waiting on the phone.

There are two challenges which make the task of finding a flight frustrating and time consuming for a travel agent. One challenge is the method the flight information is displayed and the other is the limitations in accessing the flight information.

Currently, flight information is displayed in a long, plain text list with many inconsequential details [see Figure 1]. For each flight, information such as the flight's number, gate and the number of seats for each class of service available is displayed even if these details are not important to the travel agent during the booking process. When a travel agent looks for a flight for their customer, they must sift through multiple pages of flights and often complete multiple search queries if the needs of the traveler are not met with the original resulting list of flights. Looking through pages of flights for multiple queries is time-consuming, inefficient and travel agents can miss flight options for their travelers.



```
YVR-SEE TX*YVR RE 5-15.00 FEE
 24APR  MON   YVR/PDT     YYC/MDT¥1
1WS     286 Y4 V4 B4 Q4 L4 YVRYYC 1300 1520 73W 0 X67
            M4 X4 P4 K4 S4
2AC     212 J9 C9 Y9 M9 U9*YVRYYC 1300 1523 320 S/ 0 DCA /E
            B9 R9 H9 K9 V9 N9 Q9 G9 L9
3AC/** 8310 Y9 M9 U9 B9 R9*YVRYYC 1330 1553 CRJ 0 DCA /E
            H9 K9 V9 N9 Q9 G9 L9 P9 A1
4WS     562 Y4 V4 B4 Q4 L4 YVRYYC 1210 1430 73W 0 X67
            M4 X4 P4 K4 S4
5WS     168 Y4 V4 B4 Q4 L4 YVRYYC 1400 1620 736 0 X67
            M4 X4 P4 K4 S4
6NZ/AC 4712 C4 D4 B7 H7 M7 YVRYYC 1200 1423 321 S/ 0 DCA /E
            Y7 Q7 V7 W7 T7
INTL ONLINE CONEX/STPVR TFC ONLY
* - FOR ADDITIONAL CLASSES ENTER 1*C
```

Figure 1:Current travel agent display of possible flights

Accessing the flight information is another challenge in finding flight information. Searches are limited because all flight information is distributed through Sabre, a global travel distribution system, and is retrieved through a network connection.

One of the limitations of Sabre is the type of queries which are able to be performed. A travel agent can either search by availability (which flights have seats available to book on) or by low fare (return the cheapest flights). With an availability search, all flights are returned without the price of the flight. A flight must be selected from the flight listing and then priced. If the price does not meet the traveler's needs, the travel agent must start the search over. The low fare query will only return the lowest fares and might not include all travel times. These queries do not meet the requirements of the travelers who are trying to balance many different needs, such as a customer who wants to purchase a refundable ticket (class of service) for the lowest price possible because not all information is available to the travel agent. Also, to query for multiple days, a new query with a new date is needed. To search for many days is very time-consuming and is not often completed due to the time required for this.

I was able to work with BTI Canada's team of developers to develop initial requirements and possible solutions. To solve some of the issues experienced by travel agents, we propose FlightView, a flight information visualization.

narksey@cs.ubc.ca

The requirements of the solution include:

- The display must be easy to understand and quick to learn.
- The results must be returned relatively quickly (within 2 minutes).
- The display includes an overview of all flights, but also has the ability to get more information about specific flights if needed.
- A filter to reduce the insignificant information the travel agent and traveler do not need to make a decision about booking a flight.

The rest of this paper will discuss related work, a description of FlightView and high-level implementation details will be described alongside results and future work.

2 Related Work

How to display hundreds of flights is an Information Visualization problem. Due to the lack of work done in visualizing travel information, FlightView utilizes techniques in Information Visualization including focus+context and dynamic filtering to help solve the problems with displaying hundreds of flights in a single display. To implement these Information Visualization techniques, the prefuse toolkit was utilized.

2.1 Focus+Context

An interesting problem in Information Visualization is the question of focus+context, that is how can users view an overview of their data and still be able to get more detailed information. One approach to solve this \problem is the Bifocal Display which can transform and magnify different dimensions in a visualization [see Figure 2] [1]. FlightView will utilize a simple 2-dimension Bifocal Display to allow travel agents to not only analyze an overview but also highlight an area for more specific flight details for that time. It is important that travel agents will be able to quickly move over different time periods to share with the traveler all of their flight options.

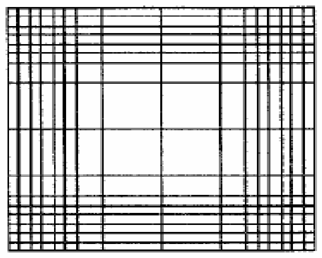


Figure 2: 2-Dimension Bifocal Display

2.2 Dynamic Filtering

The problem of viewing flight information for multiple days in an easy to understand way is similar to the problem of viewing large database queries. The benefit of dealing with flight information is that we know the dimensions ahead of time and can add filters to make the interaction easier and more successful for users.

An important aspect of viewing database information is the ability to query and filter information dynamically. In FilmFinder, this is accomplished by utilizing slider widgets on the side of the display which users can interact with to limit their queries [2]. Sliders are a useful filtering technique because users are familiar with using these to interact with other applications [see Figure 3].

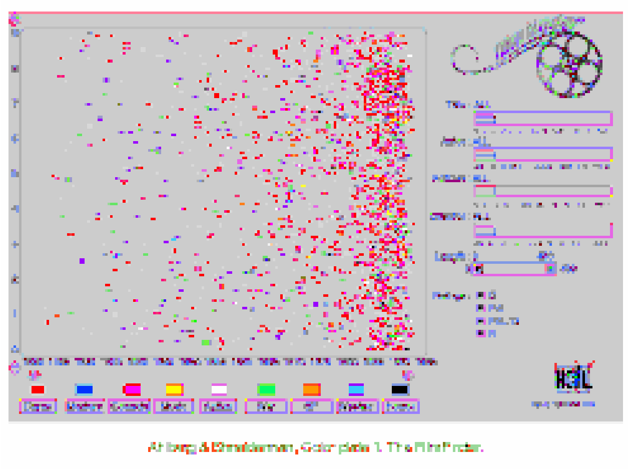


Figure 3: FilmFinder with dynamic filtering

FlightView will utilize this technique to allow users to filter their flight results.

2.3 prefuse

prefuse is a Java toolkit which utilizes Java2D to implement Information Visualization interactive techniques [3]. The toolkit enables developers to dynamically develop visualizations techniques such as focus+context techniques, force-directed layout graphs, hyperbolic trees, tree maps and radial layouts.

prefuse includes methods to easily implement visual items, layouts, graphic renders and UI controls. FlightView utilizes the these to implement the graph and the Bifocal Lens.

3 Description of Solution

FlightView displays an overview of several days of flight information, including cost of the flight, the airline and the length of the flight. The ability to view several days of information is very powerful to travel agents who, as previously mentioned, have to search through pages and pages of text to get the amount of information that can be displayed in this overview. FlightView contains a central display which presents each flight found in the search results. To aid in the interaction, FlightView also includes a Bifocal lens, filters on the right hand side of the screen, a legend for the colour of each airline and an area to display more detailed and saved flight information [see Figure 4].

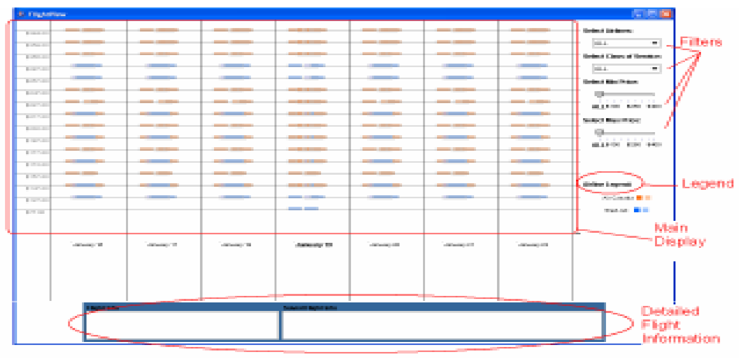


Figure 4:FlightView (larger image in Appendix)

In the central display, a rectangular colour node represents each flight. The colour denotes the airline and each airline will have a specific hue, so expert users will be able to pick out specific airlines quickly form the display. For novice users, a legend will display the airline and its corresponding colour for each airline currently in the display. Each colour selected for the airlines will need to be sensitive to colour-blindness because we don't want to hinder any travel agents performance. We ran our prototype example through Vischeck.com to ensure our colours were usable for colour-blindness. Figure 5 demonstrates what the nodes look like for all three types of colour blindness. While colours do not look the same as FlightView, the Figure does show the colours we selected were discernable from one another.

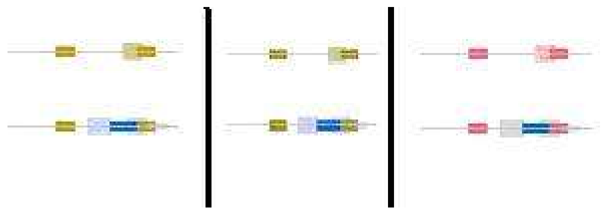


Figure 5: FlightView nodes run through Vischeck.com for Deuteranope, Protanope & Tritanope colou-blindness.

The width of the node relates to the length of the flight. The general height of the node has no specific meaning and was chosen for aesthetics and ease of viewing. If a node's start time was before the end time of the previous flight at the same cost the node's size was doubled and made 50% transparent. This was to help users be able to view nodes where otherwise they would be occluded [see Figure 6].

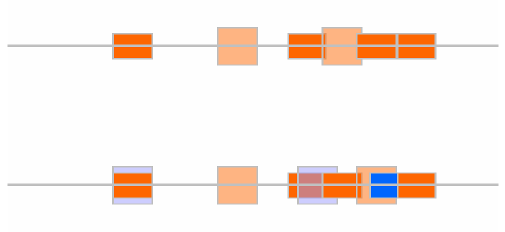


Figure 6: Example of different size and transparent nodes to deal with occluding nodes

The Y-axis is cost of the flight and X-axis is the date and time of the flight. Cost and time were chosen for the axes of the graph because these are considered two of the most important factors for travellers when booking a flight. Time was chosen for the X-axis because the width of the node can demonstrate the length of the flight. The Y-axis was chosen for the cost because users can associate low-fares to flights lower on the visualization and higher fares closer to the top and can therefore quickly spot low fares.

To compute the placement of the nodes on the Y-axis, the axis is divided up by how many different price points exist, rather than scaling the axis. This was done because there are many flights around a particular price point and scaling these causes the graph to be concentrated in a certain region and makes each individual node difficult to view.

Labels and lines for the different price points run through the nodes so if users are viewing a particular price point, they can see which other flights fall on this same price point. Bars were not added to the X-axis for each hour marker to reduce clutter and ease reading. Although, each hour is not marked with a line each separate day is marked and labelled so users can view the flights contained within a specific day.

3.1 Focus+Context

FlightView also has the ability for users to magnify and transform areas on the graph. Since there are hundreds of flights on the same day and some flight times overlap, individual nodes are hard to distinguish from one another. FlightView utilizes a simple 2-dimension Bifocal lens to allow users to highlight an area for more specific flight details [4]. The Bifocal lens transforms and magnifies the different dimensions in the graph, so areas the user moves over with their mouse are easier to read. While some areas are magnified, the rest of the graph is still viewable by the user, so they do not lose context of what day they are looking at within the graph. We also tried a Fisheye lens and a 1-dimension lens [4], but the Fisheye lens made the rest of the graph too difficult to read and 1-dimensonal lens did not magnify the nodes enough to reduce the visual clutter. Figure 7 shows an area of the graph where the lens has moved magnified the area.

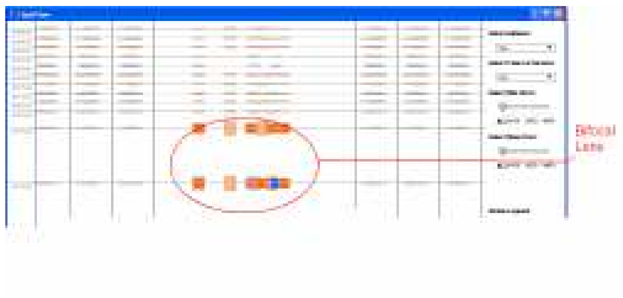


Figure 7: FlightView Bifocal lens example (larger image in Appendix)

As well as the Bifocal lens, users can get more detailed flight information by hovering over a flight's node. The information is displayed on the 'Flight Information' text area at the bottom of the screen. The detailed flight information includes, the times of departure and arrival, class of service, airline and flight number and cost.

Users can double-click on flight's node to select various flights they might be interested and want to review later. The flight's information is displayed in the 'Saved Flight Information' text area at the bottom of the screen. Once selected the, the node's colour is changed to black so users can see which flights are selected in the graph in the overview or the focused lens [see Figure 8]. This reduces the cognitive load on the travel agent when trying to remember all the possible flights and details. The 'Flight Information' and the 'Saved Flight Information' text areas are located at the bottom of the screen and not beside the node so as not to cover up any other nodes.

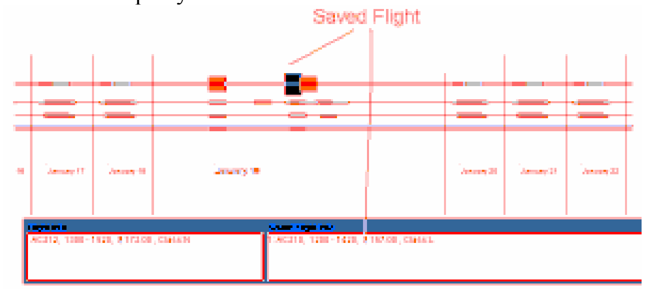


Figure 8: Saved flight information display (larger image in Appendix)

3.2 Dynamic Filtering

Other features of FlightView are the dynamic filters on the right hand side of the screen. These filters include the ability to choose specific airlines and class of services to view, and select on minimum and maximum cost of flights. For factors that are absolute, such as a particular airline and class of service users can choose what to factor from a drop-down menu. For factors with continuous values, a slider bar was chosen so users can move the values along a bar. While there are 2 sliders, one for minimum and maximum cost, in the future I would like to move these to a single slider. Figure 9 show what the airline drop-down menu looks like after the user has selected it. Figures 10 and 11 show a graph before and after a filter is applied.

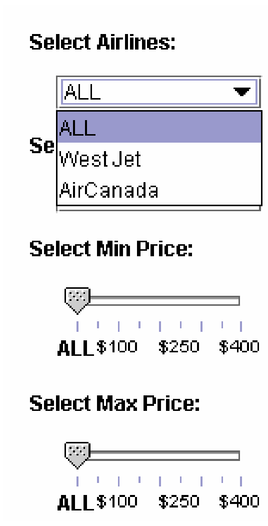


Figure 9: Example of FlightView's filters

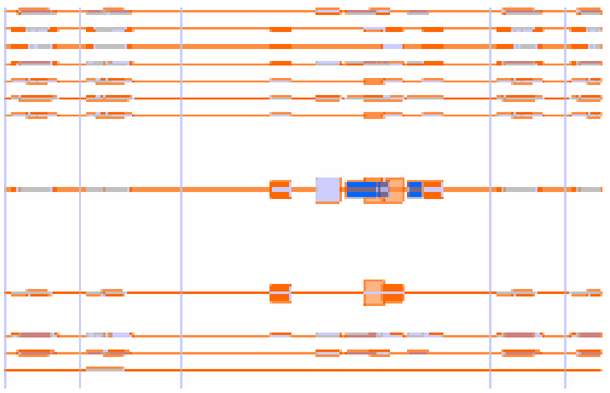


Figure 10: FlightView with no filters on data

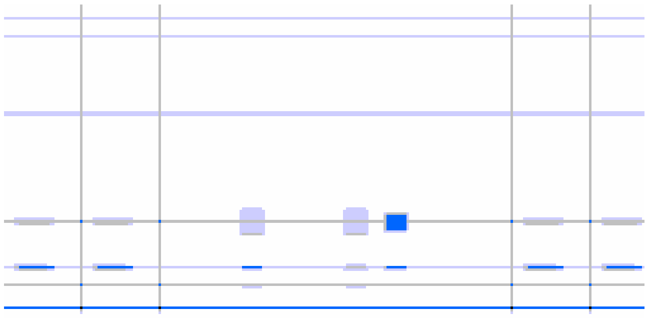
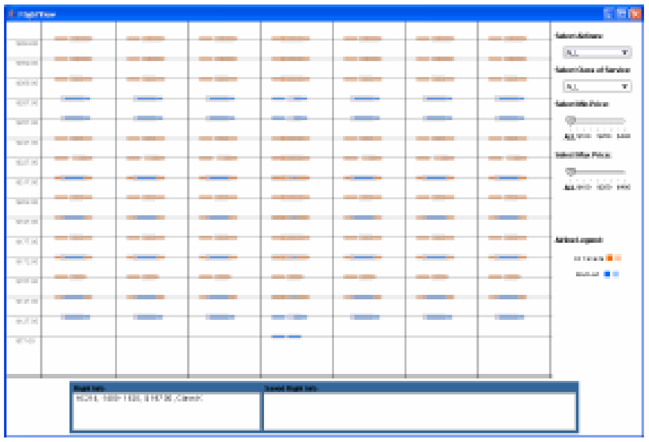


Figure 11: FlightView with airline filter on data (only West Jet flights displayed)

4 FLIGHTVIEW SCENARIO OF USE

A traveler calls into BTI Canada to book flight from Calgary to Vancouver, for business. The travel agent asks the traveler what day he would like to fly. The traveler gives the travel agent the requested date, January 19, 2005. The traveler states they must arrive in Vancouver before 10:00 am, but they can only spend $80 on the flight. The travel agent inputs the dates and the origin and destination into Agent Interface. The travel agent asks the traveler if they have a preferred airline they would like to travel on. The traveler replies that it doesn't matter as long as he gets there on time. FlightView opens up [see Figure 12].



312: Scenario of Use: FlightView displays flights for YVR to YYC on week surrounding January 19

The travel agent reviews the information returned and sees there is a flight available for that day for that prices and hovers over the flight to get more details [see Figure 13].

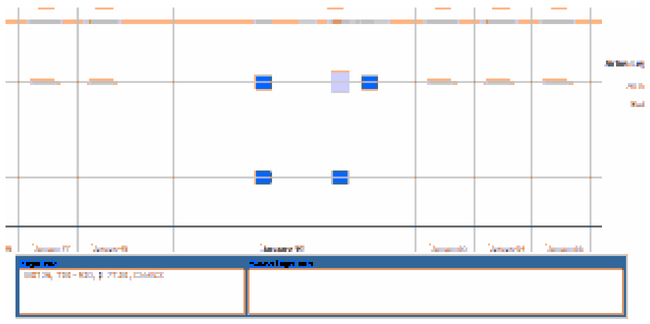


Figure 13: Scenario of Use: Flight expanded and information displayed at bottom of the screen

The travel agent informs the traveler there is a flight for the morning. The traveler states he would like to book the flights; the travel agent books the flight by double-clicking in the flight and ends the file.

5 High-Level Implementation

A FlightView prototype was developed as a proof of concept and to be able to get user's initial feedback. FlightView was implemented in Java utilizing the prefuse toolkit [3].

FlightView currently does not consume live flight information from Sabre because during development there was no access to a connection to the Sabre network. Instead real flight information was gathered and saved in a text file read in by FlightView. The prototype was implemented with flight information for the routing Vancouver to Calgary. We chose a simple domestic routing for the prototype because it there are many flights throughout the day, with different classes of service and price points, but we would not have to deal with connections and layovers. This routing is a popular routing for business travelers living on the West Coast and is representative of what type of routing would be input into FlightView. In the future, connecting FlightView to Sabre's network is necessary.

6 Results

The FlightView prototype proves this is a viable solution to build an interactive display with hundreds of nodes of flight information. This example contains 627 nodes and there is room for some scalability. Besides the number of nodes, FlightView also meets performance requirements. User feedback is shows potential and users offered many possible improvements.

6.1 Performance

The time required to return results to the user is a considerable concern for FlightView. There are two areas where time could be an issue for FlightView.

One area of concern is the time needed to render the graph. After implementing the prototype, it is clear that after the graph is loaded with this amount of data FlightView can react in real-time. When users are filtering the graph, the results are displayed in close to real-time, so the interaction with the graph is seamless.

The other performance concern is the amount of time needed to get the flight information from the Sabre network. Since Sabre's queries are not built for the type of data searching needed to display the cost for all available classes of service for several days, many different queries have to be completed and compiled to get this information. Initial time trials show the flight information for 7 days can be returned in less than 2 minutes.

There are some techniques that can be utilized to help decrease this performance time. One is to store popular routings in the morning in a database and have FlightView query the database which has a much quicker time to return search results than querying the Sabre network. The other technique to help minimize the performance time of Sabre's network is to save search queries on the user's computer rather than write them to a database. Instead of performing a new search query for the same routing, a previous query for the same routing can be returned. This will help reduce the waiting time for popular routings a travel agent looks up several times throughout their day.

Although FlightView's required time needed to return results within a certain time frame is an important performance issue, the travel agent in the long run is hopefully saving time and providing better customer service by providing all options to their travelers. However much time it takes for FlightView to display results to an agent, the manual process of looking up all those flights would always take much, much longer.

6.2 User Feedback

After the FlightView prototype was implemented, 7 users at BTI Canada were asked for their feedback. Since this visualization technique was a novel way of viewing flight information, we wanted to get user's initial impressions of the tool, rather than doing a formal study or evaluation. Their feedback will be used for the next design phase so more formal user evaluations can be completed.

After describing a brief task scenario and description of FlightView's functionality, users were asked what they liked, disliked and if they had any ideas for improvements of FlightView.

Users were generally positive about FlightView. Most could see the benefits of viewing such a large amount of flight information in a relatively small amount of space. Being able to view the price for all different classes of service for a flight was acknowledged as very useful. Also, users liked the capability of being able to pick out low-priced flights over several days quickly.

Below is a list of other functionality users liked in FlightView:

- The lens helps view specific times of flights and decrease visual clutter of flights spaces close together.
- The filters on the side to be able to narrow results dynamically. Also, being able to go back to the full view quickly and as easily as filtering the results to begin with.
- The ability to view more specific flight information when you moved over the flight's node.

There were some details users found confusing or didn't view as useful. Below is a list of functionality users generally disliked in FlightView:

- The rectangular node representation of a flight is not intuitive.
- The time on the X-axis need more labels and lines for specific times throughout the days so it is easier for users to see the hour the flight departs and arrives.

- The different size of the nodes that are overlapping is confusing and it is not clear why the nodes would be different sizes to the users. Some users thought the different size had some significant meaning, although they didn't.
- The specific flight details displayed at the bottom of the screen are too distant from the node and it is difficult to move your attention from the node to the information.
- There is no visual connection between flights selected and the flights on saved flight list so it is difficult to remember which flight information corresponds to which node.
- You have to use the mouse interaction to view flight information. The current interface travel agents at BTI Canada use is keyboard driven so users do not have to move their hands off the keyboard.

Users also had some ideas on how to improve FlightView. Below is a list of ideas for improvement:

- Change the X-axis to cost and the Y-axis to time.
- Add the ability to view queries with more than 1 condition. For example, filter out all classes of services except for X and J.
- Add the airline logo or the airline two letter code to the node when it becomes expanded so users do not need to look at the legend or the flight details to know the airline.
- Select how many days of flight information to view.

7 DISCUSSION

Discovered from the user feedback and evaluation of FlightView there are some strengths and weaknesses which will be discussed below. Also, from the user feedback we identified another group of users who may also benefit from FlightView. Lessons learned and future work is also discussed.

7.1 Strengths

FlightView is an easy-to use, display which has the capability of viewing hundreds of flights over several days. Some of its main strengths include the amount of information available in one screen, the ability to look at an overview or view more details of the specified flight and the dynamic filter to reduce irrelevant information.

7.1.1 Interactive Information display

From the user feedback one of the biggest strengths of FlightView is the ability to view and interact with so much flight information in such a small space. There is no other tool like this available to travel agents and it can save time when trying to search over several days to meet customer's needs. FlightView can help users spot cheap flights out of hundreds available without having to manually complete many queries for different times and days.

7.1.2 Overview and detail

Another feature users liked was the ability to move from the overview to more details of the flight information. The overview is useful because it allows users to quickly and easily pick out inexpensive flights over several days. The Bifocal lens lets users view more a magnified display on specific times so they can better see the times of the flight. Also, by hovering over a flight users can view the specific details of the flight including the airline, flight number, class of service, cost and time of the flight. From this users will have all the required information to book the flight.

Although, some users complained that FlightView requires mouse interaction, rather than just keyboard shortcuts, this is something which needs to be further investigated. One of the GOMS models will help quantify the amount of extra time this will require [5]. Since agents would only have to move their hand to the mouse once in every transaction this time hit might not matter. Further studies will have to be completed to see if this amount of time has a significant effect on user's productivity.

7.1.3 Dynamic Filter

Another strength of FlightView is the dynamic filter. This is a quick way users can reduce the returned set of flights to find the flight their customer is looking for. The widgets to do this are common widgets users would have used and seen before in other applications. The filtering is done in real-time so users can view their results right away and make the connection between the filters and the flight nodes.

7.2 Weaknesses

From user feedback, a few weaknesses were discovered in FlightView. These include the representation of the flight, lack of information between saved flight and node and the actual cost of the FlightView implementation.

7.2.1 Flight Representation

One issue that came up through the user feedback sessions was the representation for the flight, a rectangle, was not intuitive to users and was not a good representation. One user commented that the display looked like a "DNA sequence".

Flight which times were overlapping to users were slightly larger and transparent so the start and end time could be seen. This was confusing to users because they thought this had a different meaning than smaller and darker nodes.

7.2.2 Lack of Coupling Between Flight and Information

Adding to the confusion of the flight representation, there was a problem with the coupling of the flight and the detailed flight information. The flight information is displayed at the bottom of the screen and although this was done to increase visibility of other flights, it was time consuming to continue to look from where the mouse was to the bottom of the screen for the details.

Adding to the confusion between flight and detailed information, there was no coupling between the saved flight information and which flight was selected. So, if users saved 6 different flights the list would display this information, but users would have to search for which flight the information was for. One of the suggestions from the users discussed above was to move the flight information to appear beside the node and this technique might limit this weakness because all flight information will be displayed beside the node.

7.2.3 Cost

Another issue which came up in the implementation for FlightView is the cost associated with doing this many queries in Sabre. Each of these queries only costs $.007, but added up this can be a large expense to the company and since the travel agents don't usually perform this comprehensive of a search this would be an additional cost to the company. The benefit though is that customers will be given truly the best choice out of several days for their flight and with this added functionality and level of service, the company could benefit in the long run. Also, some of the suggestions in the 'Performance' section to improve the running time would also help to reduce the cost of FlightView.

7.3 Users

Another issue discussed in the user feedback is the type of travel and user FlightView would be useful for. While FlightView was originally designed for corporate travel agents, it became clear that it is more useful for situations where the traveler is driven by price and not by availability and this is only a subset of customers for corporate travel agents.

One of the major lines of business at BTI Canada is the booking of travel by redeeming points. FlightView may be better suited for this group of travel agents who mainly deal in leisure travel where customers are mostly concerned with the cost. This line of business also has a higher turnover rate than corporate travel agents so having a tool where users don't have to learn the details of Sabre might improve training time for new agents.

Another group of users FlightView may be useful for is users booking travel for themselves using an online self-reservation tool. Users in this situation might become frustrated searching for a flight in the traditional method because of their lack of expertise in the travel domain and time required for them to find the 'right' flight. By returning all flights over several days users might be more confident they made the right choice and therefore be happier with the self-reservation tool. Below is a scenario of use for a traveler booking their own travel using a self-service reservation system.

7.3.1 FlightView Scenario of Use for Traveler Booking Travel Online

A traveler wants to book a flight from Vancouver, BC to New Your, NY. The traveler opens up their web browser and logs on to their self-service reservation tool. The traveler inputs the origin, destination and preferred travel date. FlightView opens up and the traveler can view possible flights for several days before and after the preferred travel date. The traveler is a frequent traveler and collects travel points on Air Canada, so he filters all other airlines. He also selects business class service because he only prefers to fly business class. The only flight in the morning of his preferred day of travel will not get him there early enough to make a meeting, but he notices there is a 'red-eye' flight the night before. He slides the lens over to the previous day's flights and looks at the information for that flight. The traveler books this flight.

7.4 Lessons learned

Many lessons were learned during the implementation of FlightView, including learning a new toolkit, working within the travel domain , some iterative design lessons and designing for expert users.

One challenge when implementing FlightView was learning the prefuse toolkit in a way that would be both useful and efficient for this project. After learning the main pieces of the toolkit, it was easy to use and effective. I would recommend this toolkit for other interactive Information Visualizations.

The other lesson I continue to learn is how complicated the travel domain is. I have a Computer Science background, but spent a year working at BTI Canada. Although, I learned a lot during my year there it seems there is always new things to learn. It is complicated by the fact that there is a limitation in the functionality and services available by Sabre. Since I was limited by this, to build the visualization I had to get the information I needed by doing inefficient search queries. The limitations of working with Sabre help show the necessity of FlightView. FlightView abstracts the details of Sabre and can take the information retrieved from Sabre and give it to users in an easy-to-use effective tool.

Another lesson was in time-management. I would have liked to have had more time to complete one more design cycle, make some changes and then demonstrate FlightView to more users. The feedback I received was invaluable and would have liked to get more feedback throughout the process, not just at the beginning and end. One reason this was not done, was due to time constraints, but I would recommend doing less of something else to make room for talking to users more.

One of the most interesting things I learned was how difficult it is develop a new information display for a group of expert users. These users become proficient at using systems they are used to and might be resistant to new displays, especially if the benefit is not apparent to them. Having buy-in from a group of users and having them more involved in the brainstorming and development sessions might help this process.

7.5 Future work

The next step in this project is to take some of the suggestions from the user feedback sessions, implement them and complete formal user evaluations.

Some of the changes for the next iterative design of FlightView include:

- Moving the flight information detail to a context text box beside the node so users can quickly view the flight's details
- Removing the 'Saved Flight Information' because it takes up valuable space for the display and no users commented on the usefulness of this display. Once the user clicks on the node, it will change color so if they would like to 'remember' a node they have viewed they can quickly see the nodes that have changed color.
- Adding the airline logo to the expanded flight node to help user identify airlines without a looking at the legend.
- Adding the ability to filter by more than 1 condition.
- Unify the cost widgets into one slider bar rather than two.

There are still some difficulties without clear answers about the visualization. Some of these open-ended questions include:

- What shape should the node be? The current rectangle shape is not intuitive to users and there might be a better shape suited for this. For instance an arc might be more appropriate or a glyph which both can be shapes associated with travel. If arcs are used to represent flights techniques such as EdgeLens, an interaction to deal with occlusion [6]. could be used to deal with occlusion.
- Should the time and cost stay on the current axis?
- Is there a different interaction technique so users would be able to use the keyboard or the mouse to get more information about the flight?

After another iterative design prototype is developed, formal evaluations need to be completed to determine the effectiveness of the tool. We would like to look at travel agents using both FlightView and their traditional booking software and look at times to complete tasks and if possible measure customer service to see if the FlightView tool can help travel agents give better customer service. Also, we would like to investigate the effectiveness of FlightView for both beginner and expert and users other than travel agents in corporate travel, such as travel agents in leisure travel and travelers booking their own travel online

8 Conclusion

I worked with BTI Canada, a corporate travel agency to develop an Information Visualization which displays and filters days worth of flight information. The implementation was completed using the prefuse toolkit and utilized Information Visualization techniques to help solve the focuc+context and dynamic filtering problem. User feedback showed that users were generally happy with the interactive visualization and offered many areas for improvement. More work needs to be completed to improve the effectiveness of FlightView and user studies need to be completed to prove the viability of this solution.

Referernce

[1] Y.K. Leung and M.D. Apperley. A review and taxonomy of distortion-oriented presentation techniques, *ACM Transactions on Computer-Human Interaction*, Vol. 1, No. 2, June 1994, pp. 126-160.

[2] Christopher Ahlberg and Ben Shneiderman. Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. In *CHI 1994, Human Factors in Computing Systems*., 1994

[3] Jeffrey Heer, Stuart K. Card, and James A. Landa. prefuse: a toolkit for interactive information visualization. y. In *CHI 2005, Human Factors in Computing Systems*, 2005.

[4] R. Spence and M. Apperley, Data base navigation: An office environment for the professional. *Behaviour and Information Technology*, 1(1):43--54, 1982.

[5] B.E John and D. E. Andkiera1996a. TheGOMSfamily of user interface analysis techniques: Comparison and contrast. *ACM Trans. on Computer Human Interaction. 3*, 4.

[6] Nelson Wong, M. Sheelagh T. Carpendale, Saul Greenberg, EdgeLens: An Interactive Method for Managing Edge Congestion in *Graphs Proc. InfoVis03,* pp 51-58.

Appendix

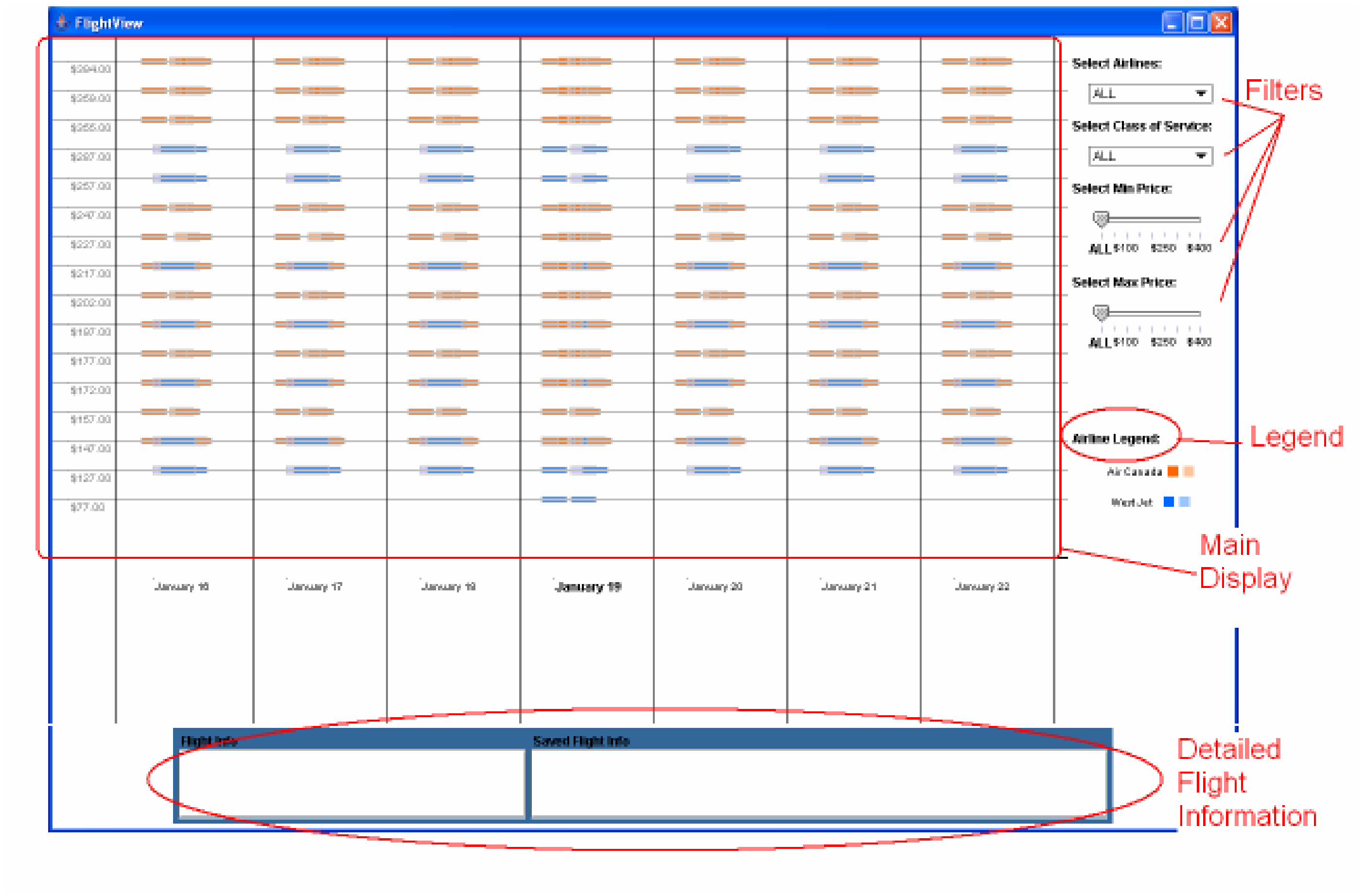


Figure 14: Larger Overview Image

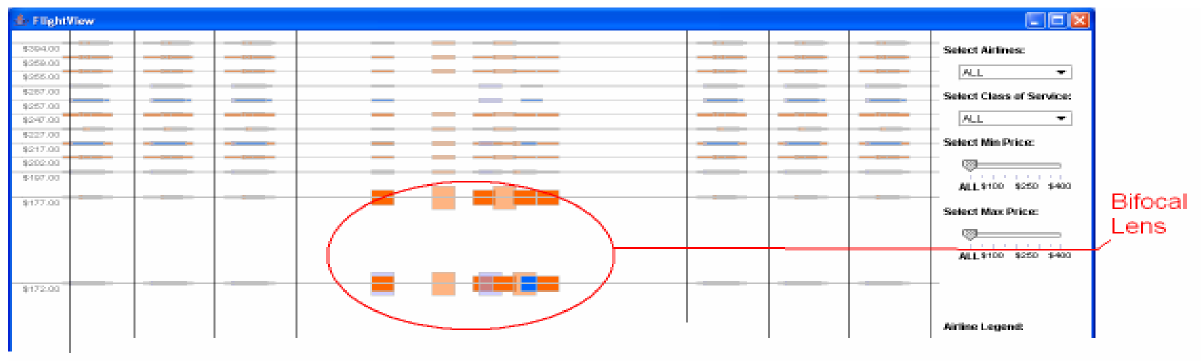
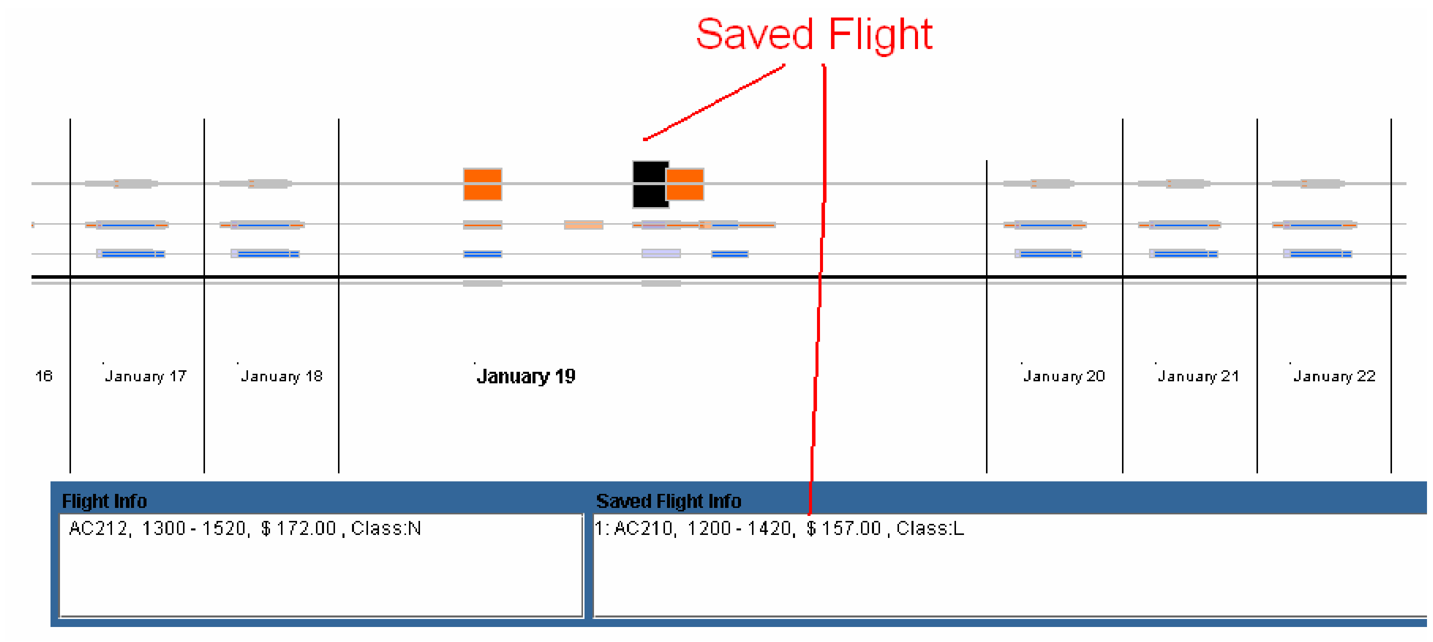


Figure 17: Larger lens image



AC212, 1300 - 1520, $ 172.00 , Class:N

1: AC210, 1200 - 1420, $ 157.00 , Class:L

Figure 8: Larger Image of Saved Flight Information