Team Cancer: Rose Kirby, Christine Lu
March 2, 2011

Information Visualization - Application Project on SEER Cancer Data

Motivation and Background:
For this project, we decided to use the National Cancer Institute’s Surveillance, Epidemiology and End Results (SEER) database to supply information to our application visualization project. The SEER database is an comprehensive collection of medical records and cancer statistics in the United States between the years of 1973-2007. It contains over 6 million case studies, each with associated patient information like age, race, gender, year of diagnosis, state and county of residency, lymphoma subtypes, survival time, and countless other options. Because the data is so vast and extensive, it is difficult for anyone to peruse the database and gain any particular insights without the aid of visualization tools.

We started by using the SEER*Stat program to search through and filter information in the database. Seer*Stat itself does not provide any visualization capabilities, but allows us to view and export smaller sets of data by providing query options. However, the size of the data made it impossible to load into some programs, Treemap and HCE3 for example, in its entirety. We then decided to pare the information down to smaller values, and worked with a smaller set of 40,000 case studies, from the years 1998-2007, and a set of 10,000 case studies from 1973-2007.

Results and Visualizations:
**Fig 1: Division of cancer types (1998-2007)**
This treemap shows the different sites of cancer: each type is split into regions for gender. The size of each cell is determined by how many people are affected. By far, the two largest categories are breast cancer and prostate cancer, despite the fact that they are both limited to affecting only one gender.
Fig. 2: Exploring gender differences: Survival rates of men and women by year of diagnosis (1973 - 2007)

In the past, general medical research and drug tests were performed on male subjects - women introduced monthly hormonal variables that may have interfered with the results. Furthermore, between the years of 1962 and 1993, the FDA’s strict guidelines excluded fertile women from drug trials, in the chance that their participation would inadvertently influence a pregnancy or a fetus. We were interested in seeing if this resulted in a long term change for female survival rates before and after the NIH Revitalization Act of 1993 - which set guidelines for inclusion of women and other minorities in clinical trials.[4]

As a standalone visualization, this graph doesn’t show us too much. But when considered in the context of its background, it shows that the inclusion of minorities has not yet shown a significant difference in cancer research.
Fig 3. Refusing treatment yields second highest average survival time (1998-2007)

This bar graph shows the various radiation treatments available. The last bar indicates that those who refused lived for an average of 5.23 years - making it the second highest option after “Combination of beam with implants or isotopes.”
Fig. 4: A closer look at treatment and survival rates (1998-2007)
This line graph takes a closer look at the information from Fig. 2. Each line represents a different year of aggregated data. From this chart, it is evident that while refusal of treatment can lead to a higher average survival rate, it can’t reach the same peaks of the riskier treatments.
Fig. 5: Race and responses to treatment options
African Americans respond better to most types of radiation treatments than other races.

Fig. 6: Year 2000 shows marked increase in quantity of ovarian cancer diagnoses (1998-2007)
This bar chart shows the number of ovarian cancer diagnoses per year. Each year is subdivided based on the grade of the cancer.

Fig. 7: Year 2000 shows marked increase in quantity of prostate cancer diagnoses (1998-2007)
This bar chart shows the number of prostate cancer diagnoses per year. Each year is subdivided based on the grade of the cancer. The number of diagnoses has steadily declined since 2000.

Fig. 8: Lack of clear trend in quantity of brain stem cancer diagnoses (1998-2007)
This bar chart shows the number of brain stem cancer diagnoses per year. Each year is
subdivided based on the grade of the cancer. The number of diagnoses has oscillated unpredictably each year.

Program analysis:
We followed Shneiderman’s “Eight Golden Rules of Interface Design”: http://groups.drupal.org/node/8248

- SEER*Stat:
  1. Consistency: Some sessions (filtering options) will allow the user to select multiple traits at a time, while at other times, while others require the traits to be entered one at a time. The frequency session only allowed you to search for 5 columns at a time, while the case listing session allowed you to select variables to your heart’s content.
  2. Shortcuts: There are very few shortcuts in SEER*Stat; most of the filtering and searches have to be done manually, and even the most common functions (like “bring up a new session window”) do not exist.
  3. Informative feedback: A help function does exist, but it does not provide feedback on what you are doing as results pop up. It can get difficult understanding what information has been pulled out of the database.
  4. Closure: The sequence of actions are organized into groups: the beginning (where you bring up the session filters), the middle (where you define your query), and the end (where you get results). It serves as a clear indication that we can prepare and move on to the next course of action.
  5. Error handling: When the user makes mistakes (for example, including more columns than the database can handle), SEER*Stat will tell you exactly what you need to change. Fixes were easy to make.
  6. Easy reversal: The program failed in this respect: the dataset must be closed in order to execute another one, so unless a backup is made, the information from that session is lost.
  7. Internal locus of control: The control was very much in the user’s hands; each event is precipitated by an action of the user. To filter results, the user provides a boolean statement detailing which variables, which years, etc that they want to see.
  8. Short term memory load: When the data is loaded into the table, it’s hard to tell what each variable (in the column headers) mean. In addition, certain functions and operations were split into separate windows. The short term memory load on the user was quite high.

Suggestions:
- Allow combined databases. Inside seer, they provide separate databases such as “incidence” or “expected survival rates”, but there is no easy way to consolidate them.
- Allow users to see a preview of their data without requiring they load the entire data set - with over 6 million entries, even a erroneous search can take a long time to execute.
- Allow users to specify that they want X random entries from the query. There were
usually too many entries in the database to be used in the information visualization programs... as such, the user may be limited by other software capabilities. Again, this can save time.
- Have some sort of internal visualization tool. Even a simple one can be extremely effective, since other programs, like Treemap, HCE3, and Spotfire can’t import as much information as exists in the SEER*Stat database.
- Include defining words in a faster way. The definitions are often hard to keep track of, users can greatly benefit from a faster way of pulling up the definitions for terms. A possibility is allowing the user to double click on the word to pull up its meaning.
- Allow the user to execute the search in the same window they construct the query. The lightning bolt for “execute”(search for data) currently stays on the session bar of the main program while the user constructs queries in a separate window. Nowhere in the session windows does it allow you to actually see the results - not even in the “output” window.

**Summary:** The SEER*Stat program is not an intuitive program, and working with the raw data itself may be a prohibitive obstacle to those weak in computer science. Not only is it a tricky interface to use, a lot of the information inside is not well explained.

**Spotfire:**
1. Consistency: Spotfire’s consistency is good, mostly because no extra labelling is used. This helps new users acclimate to the system, and optimize the cost of learning time.
2. Shortcuts: We found Spotfire’s shortcuts very useful, though an annotated help for learning/scrollover would be a useful addition. We would also appreciate more scrolling/scrollover on the actual visualization for setting the visualization’s properties.
3. Informative feedback: There is consistently a large amount of feedback being provided to the user, but there distinction cause and effect chain is not always clear.
4. Closure: There is not a clear set of actions to create a visualization so there is no sense of visualization completion. Even as you play with the data, it’s very easy to change a good visualization to something else to see if you can get something better. There’s no clear end point.
5. Error handling: Spotfire will automatically exclude data ranges not present between columns, and will provide appropriate warnings when the dataset is too large to be imported into spotfire. There should be an option, however, to go further into why certain data or property options are excluded, because once an option is excluded, it can be difficult to ascertain why the option is not currently present and how to bring that option back.
6. Easy reversal: Spotfire has a useful undo function that can allow you to return from any step if something happens. However, when working with large sets of data, the response time is too slow.
7. Internal locus of control: This is one of Spotfire’s great strengths: the degree of user control over the program and its outputs is such a great tool. However, the learning curve can be significant here.
8. Short term memory load: The program provides a series of redundant information to
allow the user to refresh things in short term memory.

Suggestions:

- Allow the user to select location and size of labels and axes. They sometimes went off of the visualization area when using a smaller screen size.
- Provide an option so changing between visualizations for the same page/data set doesn’t change the parameters you choose to look at. The different screen sections were all based on parallel data, but sometimes the user wants to display multiple visualizations based on different data sets at the same time.
- Provide an option for maps based on common queries: world map, map of the United States, etc. Requiring that the user input coordinates is not as intuitive or easy as allowing the user to provide strings with state/country names, i.e. “New York”.
- Allow user to truncate data or pick a random sample from the set if it is too large to import in its entirety.

Summary:
Spotfire is a great program and tool for the average person to visualize information and get insights. It is fairly easy to learn, but time for the learning curve must be factored in to initial use difficulty. While we have a few suggestions, it was an informative and useful tool.

Conclusions:

There is a lack of support for very large data sets on both the end of the data set providers and the visualization tool makers. Spotfire slowed drastically when dealing with a huge data set. The SEER registry does not provide strong interfaces for exporting large data sets, especially in ways that can be used by visualization tools. Additionally, systematic exploration methods for dealing with large sets of data need to be developed. The current method of generating each visualization individually works fine with limited data sets but is a limiting factor when data sets include large numbers of parallel but distinct quantities that would be best viewed as a large set of visualizations differing on a single variable. Users should not be slowed by the process of creating these large, though only slightly differing, data visualizations.

Citations and Previous Works:

Previous Information Visualization Class Projects at UMD:
