A Visual Interface for Product Exploration on Online Shopping Websites

Cameron J. Smith

March 16, 2017

Abstract

Most online shopping websites provide a customer interface that simply lists all products available to the user. The user can then scroll up and down to explore different options. Everyone that has shopped online is familiar with this interface type. However, there are drawbacks to list interfaces. There is no obvious relationship between any two products that are displayed right next to each other. This means that during exploration users cannot use the position of a product in the list to draw information about that product. Is there a different type of user interface that is more helpful for shopping for products? This paper describes the implementation and evaluation of a novel two-dimensional visual shopping interface which arranges products on the display by similarity. The interface consists of an undirected graph with each node of the graph representing a different product. The products that are placed closer to one another can be assumed to be more similar, and those that are farther away are less similar. We propose that this new interface will be better for product exploration, because the user can now intuit information about a product based on where it is located on the screen and what products are next to it. We perform a user study to evaluate the claim that our interface is better. The results of this user study were inconclusive, however we got valuable feedback on the usability problems of our interface.

Contents

List of Figures

List of Tables

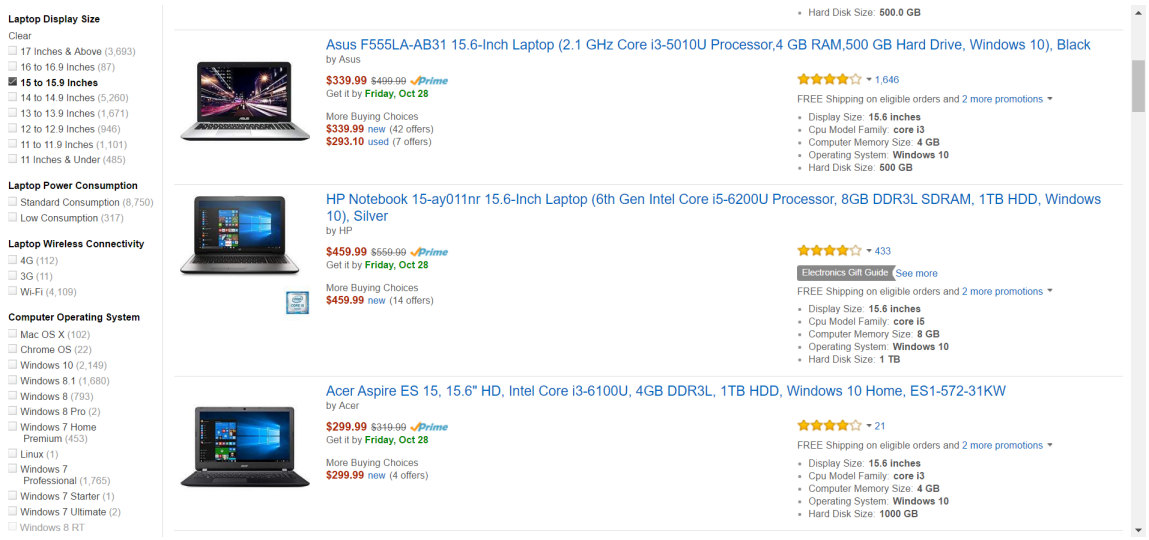


Figure 1: Standard list interface found on online shopping websites.

1 Introduction

Over the past couple of decades, consistent use of the internet has become common in the lives of a large number of people. Of all the tasks that people perform online, one of the most common is undoubtedly the task of shopping. Online shopping has some key advantages over traditional shopping. It is more convenient, because a shopper does not have to physically go anywhere to make a purchase. On top of that, online retailers have a larger selection than any brick-and-mortar store could possibly offer. With the change from a physical store environment to a virtual one, the way that people shop changes. In traditional stores, customers have the physical product in front of them. They can inspect it, compare it to other things that they see on display, and in a glance, they can see all that a store has to offer in the product they are looking for. This is not true with online shopping, especially when the selection for a product is very large. Customers shopping online in large product domains cannot easily get a good view of all their options. This makes shopping online more difficult. For this reason, it is important for online shoppers to have a good way to explore products in the domain that they are looking to buy in, because it will help them more easily find items that they will be satisfied with.

There are currently two ways in which online retailers, such as Amazon, allow their customers to explore any given product domain. The first, displayed in Figure 1, is a list interface that results from typing in a query. This type of interface tends to have filtering options on the sidebar to help narrow down what is displayed. These options can include setting a minimum or maximum price, or filtering the results on some value for some particular attribute of the product. The second way shopping sites help users explore the product domain is by suggesting other items on a product's information page. These suggestions tend to be

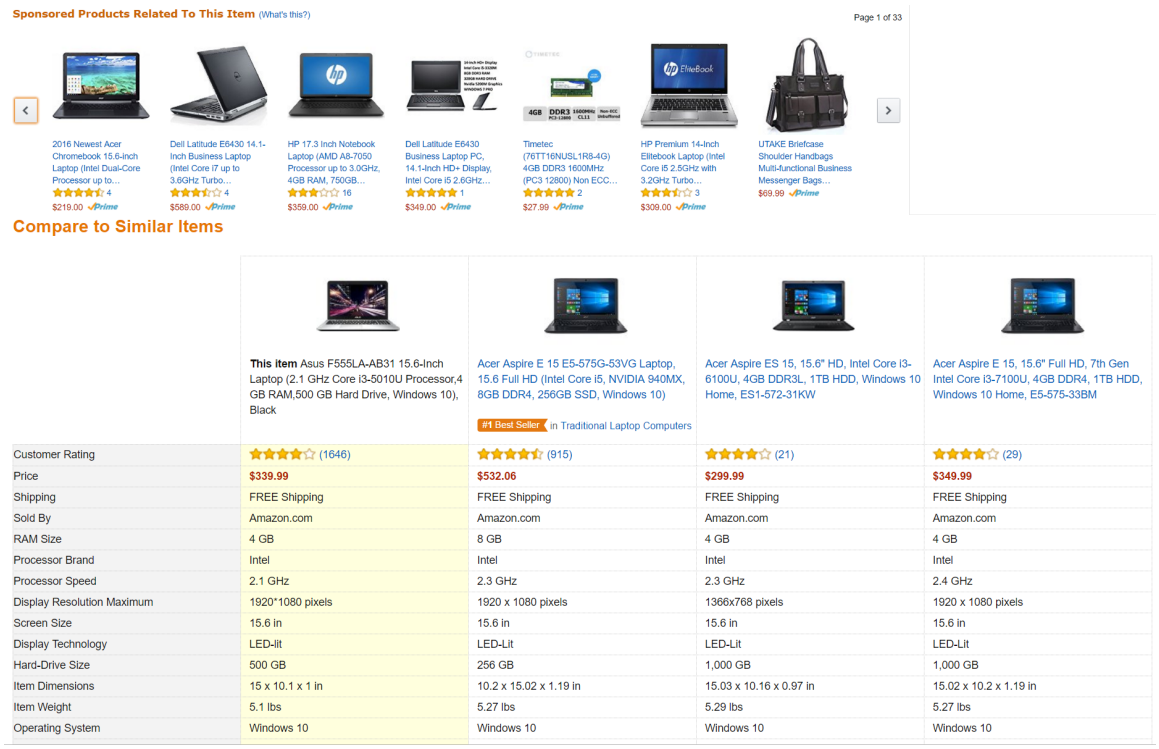


Figure 2: Suggestions for other products found on product information pages.

products that other people have either viewed or bought or they could be products that the website deems as similar to the product whose page the user is currently viewing. Figure 2 shows how Amazon does this with a "Sponsored Products Related to this Item" section and a "Compare to Similar Items" section. A majority of online retailers have sections similar to these displayed on product information pages.

For both methods of exploration, the products are presented in a list interface, where each product is displayed one after the other. For the results page list (Figure 1), all shopping websites utilize a search engine to decide the order in which the results are displayed. Each website has a different algorithm that takes many variables into consideration to decide the order. Common variables they consider are recent sales volume, number of verified product reviews, and number of clicks on the product. High values for all of these will get a product displayed higher on the list. Is this the most helpful way to organize products for a user exploring a product domain? From this method, a user can easily see what the most popular and best selling products are, but that is all they are helped to see. Aside from having a similar count of sales volume and verified product reviews, there is no guarenteed relation between any products that are close to each other in the list. From the perspective of the values of the attributes of the products, items are essentially displayed randomly throughout the list. A laptop that has a 2TB hard drive and costs 2500 dollars can be displayed right next to a 500 dollar one with a 256GB hard drive, so long as they have similar

Figure 3: Our new visual interface for product exploration. Closer products with connections between them are more similar than products at a significant distance.

values for what the search engine algorithm decides upon. It is true that a customer can use the sidebar filter to refine what the attributes are of the products that are being displayed, but even then the display is still being ordered by the same search engine algorithm. In fact, it is because the interface is a list that necessitates the display to be organized in such a way. Lists are one-dimensional, so the choice that is made in regard to how the display is going to be structured is going to be somewhat limited. The display could be ordered upon highest to lowest value of some attribute, or in some such way, but the choices are limited. What if a customer finds a product that they like and they want to see others that are similar for comparison? The list interface in Figure 1 cannot be used for this task because values of product attributes do not play a direct role in the ordering of the products in the list. The list interface in Figure 2 can be used for this task, however it is off on the information page of a product and is not the most convenient to use. Is there a different way to display products that customers will find even more helpful to their shopping experience?

This study proposes a new interactive way to display products within a domain to allow for better exploration by the user. We propose a 2D interactive GUI application, pictured in Figure 3, which will replace the results list interface as well as the similar products display on product pages. Each node in the

interface represents a product that is available to the consumer. A normal line connection between products means that those two products are significantly related in terms of the values of their attributes. Note, that there are special line connections, but we will address these in Section 3. With all similar products being close to one another and connected by lines, we know that any products present in a region are significantly related to each other. Also note that our interface has options on the left sidebar filter the products on the screen to have certain attributes, just like the old result interface. The implementation of this interface in place of the other two current methods would alleviate two problems. The first problem is a customer being on a product's page and realizing that they do not like the product. The solution would simply be to return to the explorer interface, and then navigate to a different location away from the area where that product was. Since this interface guarantees that all alike products are in the same area, then moving to a different area would allow the user to find different products. The second problem is a customer finding an item that they like, but wanting to look at similar items for comparison. Using the proposed interface, the solution is to return to the explorer interface and to look at the things in close proximity to the item they were on. We hope that a GUI system like this will offer is a more efficient and helpful way for customers to explore the options they have on a website for the product that they want to buy.

The rest of the paper is structured in the following way. Section 2 introduces other research related to the focus of this paper. Section 3 covers how we deal with the problems of building the interface. Section 4 discusses our evaluation of our interface compared to the list interface. Section 5 covers the results of the user study. Finally, in Section 6 we discuss plans for future work and the conclusions that we can draw from our study.

2 Background and Related Work

Online shopping is a phenomenon that came into being over the past 20 years due to the rise of the internet. This relatively new activity has prompted interest in researchers and companies who want to make it easier and more convenient for people to buy things. The research by Sannier, et al. [6] centers on two different methods for displaying products to help customers decide what to buy. These two methods are Comparators and Configurators. Comparators put the different values of a product's attributes in a table view with values of other products. This allows the customer to compare attribute values of different products in a fast way. However, the problem with comparators is that the table format can get much less visible when a large amount of products are being compared, and even more so when a large amount of attibutes are being compared. The other method this study examines is Configurators. Configurators allow the user to select values for the attributes that they want a product to have. The system then presents the customer

with the options that are closest to their specified value preferences. After testing both of these methods, the researchers concluded that there are good and bad aspects to both interfaces. The Comparator allows users to compare and contrast their options more. However, the Configurator allows users to narrow down their options much faster. They concluded that there are trade offs with both approaches.

Parikh and Sundaresan [3] created a system to explore products on the eBay website by brand. They realized that many people shop by brand and wanted to bring that idea of brand loyalty to the forefront in an online product exploration system. The system works by prompting the user to enter a category in which they wanted to buy something in. It then displays a variety of brands that make products for that category. The user can then pick the brand that they like best and explore the products that they are given. Their study concluded that their system was useful in that it allowed the customers to explore the products avaliable to them in a different, but just as helpful, way. Riehmann, et. al. [4], also created a product exploration system which displays all of the products in a domain through the use of parallel coordinates. Parallel coordinate representation is a way in which one can visualize a multi-dimensional dataset using only a 2D graph. It does this by displaying vertical lines to represent the different dimensions, or in the case of products, the different product attributes. Different places on these vertical lines stand for different values that the attribute can be. A curve then connects the name of a particular product to the particular values on each vertical line that make up the product. Following the curve, a user can see all of the values for each attribute of the product. The study found that a benefit of using a system like this is that it allows customers to see all the products available to them in the domain they want to buy in. The study also found that the visibility on the display can be very bad and also in the current form the interface might be confusing for public use, because not all people are comfortable with reading and understanding such a graph representation. Both of the prior two studies are of interest to us because they both show that there are different, equally viable, methods by which interfaces can present products to customers.

Bryan and Gershman [1] point out that online shopping websites are very good at supporting the behavior of search. A user can type in a term and they will be able to find the thing that they are looking for with relative ease. However, this is not shopping. Rather, it is information retrieval. The actual activity of shopping does not require the shopper to actually know what they are looking for. Instead they browse what is on display to them. But Bryan and Gershman point out that browsing is also restrictive in the sense that the customer is only ever following predefined paths that will not necessarily take them to something that they will want. Casner and Larkin [2] investigate how changing the representation of an interface changes the cognitive load that a user endures while using it. They have participants perform a task of reserving a flight using four different interfaces. They find that the best interfaces for the most efficient completion of this task are the ones that "swap visual operators in place of logical ones and prune the search through visual

cues." The first part means that representing information by means of a visual structure, color and distance comparisons.. etc, is better than trying to communicate this information through non-visual means. The second part means that graphical representations tend to arrange things such that the user can ignore some part of the information being represented so that they can find the information they want faster. They also note that graphical interfaces are not always the best option. It depends heavily upon the nature of the task itself. Some tasks are more efficiently accomplished with a graphical interface, while others are not. In our study, we believe that shopping is a task that can be more efficiently completed using a graphical interface. We follow the two conclusions of Casner and Larkin by representing similarity through distance, which allows users to ignore certain parts of the products that are displayed to them, to better focus in on products that they might like more.

3 Construction of the Visual Interface

Our goal is to construct a user interface to be used on online shopping websites, which displays products sorted by similarity. In order to accomplish this goal, we need to consider two main problems. The first is telling how similar any two products are. Note that our interface will only display one product domain at a time, meaning that if cameras are being displayed, then only cameras will be present on the interface. How do we tell how similar any two cameras are in the dataset? The next problem is how we will actually display a product domain as an undirected graph on the 2D interface. Both of these problems, and our solutions to them, will be discussed in this section.

3.1 Similarity Metric

Having a way of quantifying how similar any two products are in a product domain is necessary to sorting by similarity on our interface. To do this, we utilize a similarity metric. Specifically, we chose to use Euclidean distance. This metric is a good for our purposes, because it does a good job of ordering products by telling how far away they are from each other in an n-dimensional space of the product's attributes. We start the process of Euclidean distance calculations with a product dataset. Obviously, products have shared attributes, because they are all the same type of product. We take the values of the product's attributes. These values will be one of two types. The first type is numeric. As it sounds, these are attributes that are expressed by numbers. The other type is categorical, meaning the values are categories. To compare using Euclidean distance we need to put these values for a product in a vector. We also normalize the values of every attribute to be between 0 and 1. This makes it so that every attribute contributes equally to the

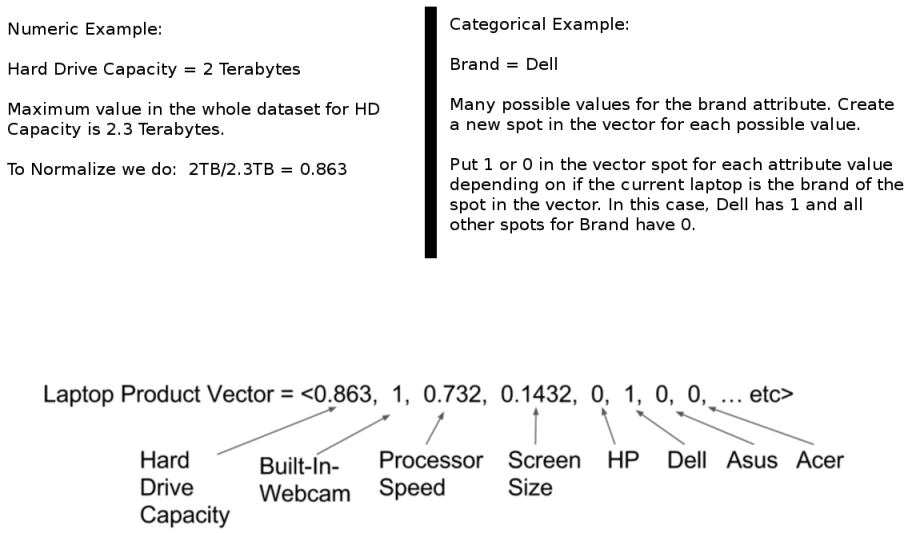


Figure 4: Numeric and Categorical examples of putting values into a vector. One vector like this is made for every product in a dataset.

similarity value. We chose to do it this way, because we felt that putting different attributes at different weights would not result in an accurate representation of true similarity between different products. To put numeric values into the vector we normalize the value by dividing it by the largest value that a product has for that attribute in the dataset. To put categorical data into a vector, we need to make a new spot in the vector for each different possible value that the attribute could be and then if the product has that particular value, we put a 1 in that spot of the vector. If the product does not have that categorical value, then we put a 0 in the spot. We do it this way for categorical data, because there is no ordering implicit in this type of data as there is in numerical data. In numerical data, we know that 1 is bigger than 0 and 2 is bigger than 1, so 2 is bigger than 0. But with categorical data, we cannot say that Red is bigger than Blue, or Yellow is bigger than Red. There is no ordering for this type of data, but having an ordering is important for using the Euclidean distance formula. We get around this by splitting up the values of categorical attributes into different spots in the vector. Figure 4 has examples of numeric and categorical values being put in the vector. Once we have each product in the dataset in normalized vector form, we can apply the Euclidean distance formula to pairings of vectors. In fact, we need to apply it to every possible pair of vectors in the dataset. This will result in numeric values representing the similarity between every pair of products. The closer the value is to 0, the more similar two products are in terms of the values of their attributes. With this we have a representation of similarity that we can now use to structure our interface.

```
-------------------------------------------------------------------------------------------
Placing Algorithm Pseudo-Code:

1. Place Product1 on the canvas and put it in a Queue(F)
2. While F is not empty
3.     Pop 1st product(p) in Queue off
4.     If spot to the right of p is not taken
5.         Find Product(a) to go in that spot, place a in spot on canvas, and push a on to F
6.     If spot to the left of p is not taken
7.         Find Product(b) to go in that spot, place b in spot on canvas, and push b on to F
8.     If spot below p is not taken
9.         Find Product(c) to go in that spot, place c in spot on canvas, and push c on to F
10.    If spot above p is not taken
11.        Find Product(d) to go in that spot, place d in spot on canvas, and push d on to F
-------------------------------------------------------------------------------------------
```

Figure 5: Pseudo-code detailing the basic flow of the main section of the Placing Algorithm.

3.2 Position Placing Algorithm

The design of the interface consists of an undirected graph where each node in the graph represents a product. Around each node, there is space for connections to four different nodes. The reason that we choose to do it this way instead of having a cloud type display is for the sake of neatness and clarity. Having a maximum four connections for each node is a lot more managable than having any amount of connections coming off any node to similar products. A user can mouseover any node in the graph and have information about that product be presented in a pop-up. The user can also click on a button in the pop-up to go to the information page of the product.

When it comes to the implementation of this interface, the most significant aspect is the algorithm that places products on the display. From where we left off in the last section, we had similarity values for each pair of products in a product domain. Before we get to the placing algorithm we need to do one more thing. For each product in the product domain, we need to find the 8 products that are most similar to it. The way we do this is by searching for the 8 similarity values for each product that are closest to zero. We then store those 8 most similar products in a list which we call the CloseProducts. To reiterate, each product in the dataset has a CloseProducts list containing the eight products that they are most similar to. This list is then used in our placing algorithm.

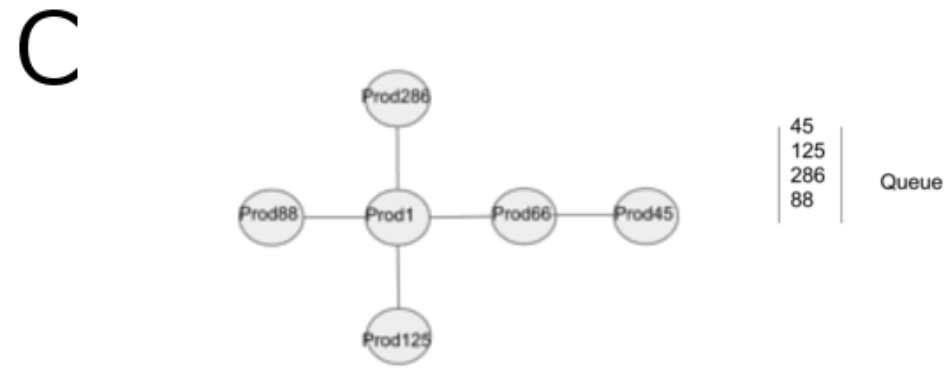
The pseudo-code for our algorithm can be seen in Figure 5. The algorithm starts out by placing an arbitrary first product onto the canvas (Figure 6-A). Then it has four spots around that node to fill with similar products. It does this by looking in the CloseProducts of the first product and taking the most similar product that has not already been placed on the display. It then puts that product that it placed on a queue, so that the open spots around that product can be processed later. It does the same thing for the rest of the spots around product 1, until they are all filled (Figure 6-B). Then it pops the first thing off the queue and starts to process the spaces around that by looking in the CloseProducts of the product that it popped off the queue and finding the most similar product that has not already been placed. It places that product

Figure 6: Snapshots of our algorithm.

Prod1-CloseProducts: Prod66, Prod88, Prod286, Prod125, Prod234, Prod277, Prod7, Prod43

Prod1-CloseProducts: Prod66, Prod88, Prod286, Prod125, Prod234, Prod277, Prod7, Prod43

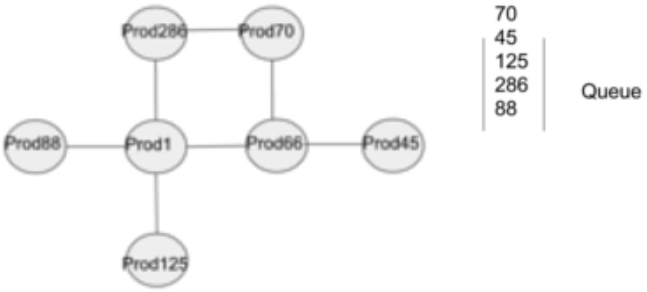
Prod66-CloseProducts: Prod125, Prod45, Prod88, Prod1, Prod70, Prod17, Prod96, Prod130



Prod66-CloseProducts: Prod125, Prod45, Prod88, Prod1, Prod70, Prod17, Prod96, Prod130

Figure 7: Snapshots of our algorithm.

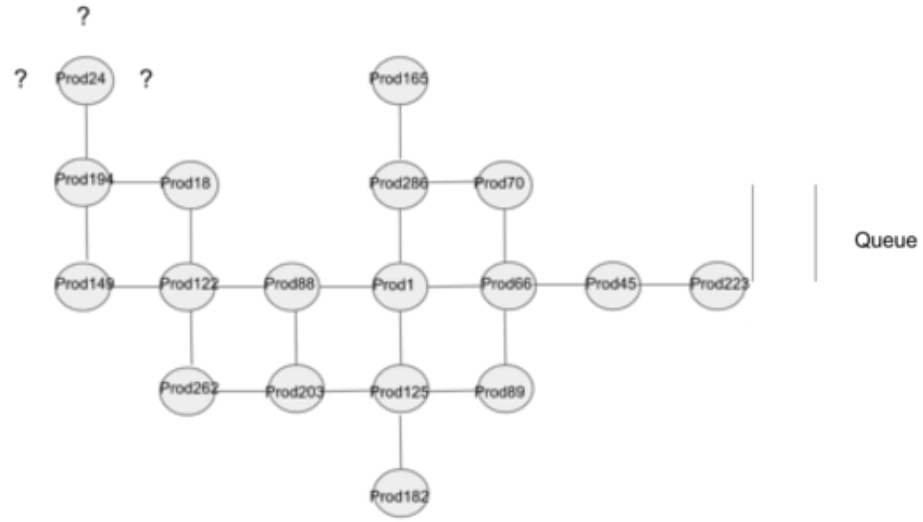
D



Prod66-CloseProducts: Prod125, Prod45, Prod88, Prod1, Prod70, Prod17, Prod96, Prod130

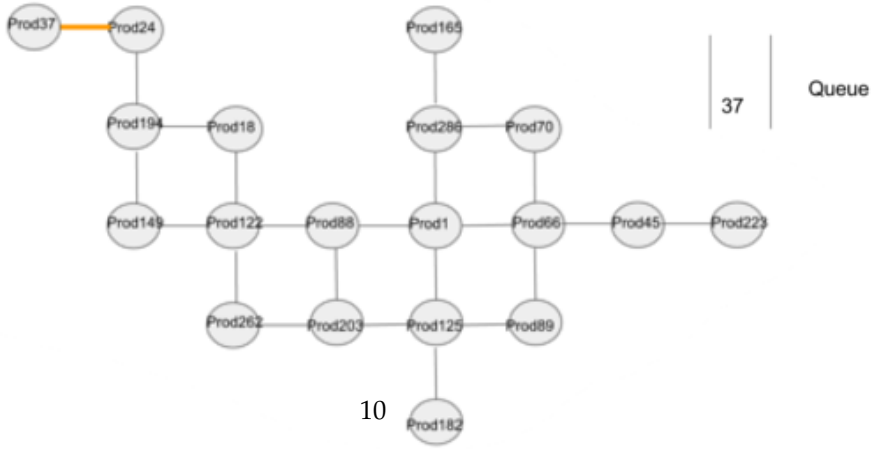
Prod286-CloseProducts: Prod88, Prod1, Prod70, Prod245, Prod165, Prod125, Prod45, Prod96

E



Prod24-CloseProducts: Prod149, Prod18, Prod194, Prod1, Prod203, Prod182, Prod70, Prod165

F



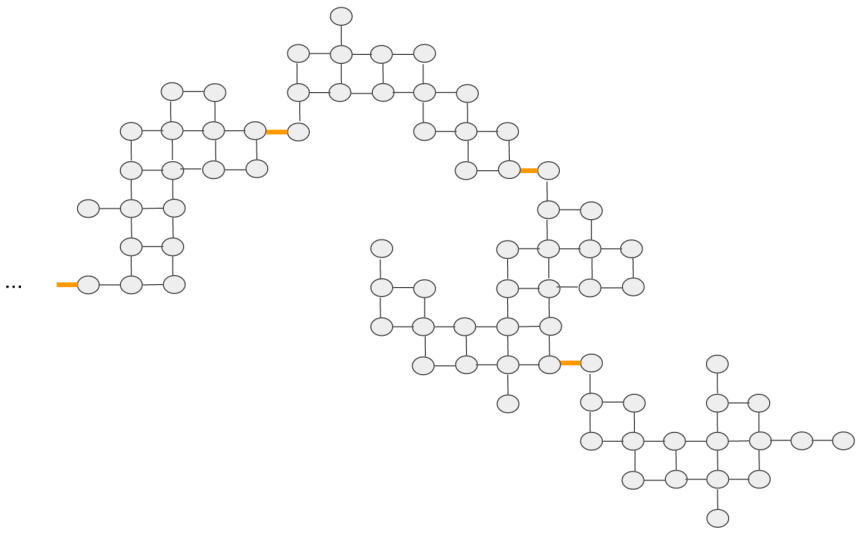


Figure 8: Every product within an island structure of connected products with normal connections are similar. Products from two different island structures are significantly different.

and puts it on the queue for later processing of the open spaces around it. The situation where an open space borders two different nodes is dealt with by going into the CloseProducts of both of the bordering nodes and finding the most similar product that has not already been placed that is in the CloseProducts of both nodes. If there is no such product, then we simply do not place anything in that spot. This allows us to draw a connection from both of the bordering nodes to the new node because both bordering nodes are similar to the new node (Figure 7-D). The algorithm continues placing in this manner until there is exactly one product left on the queue, and when that product gets popped off the queue to process the open spaces around it, there is no product left in its CloseProducts, that has not already been placed (Figure 7-E). Therefore nothing can be put around this node, but we cannot move on because there is nothing left on the queue to move on to. Instead, we go back to all of the similarity values that we calculated between all pairings of products in the dataset. We scan all the values for pairings that involve this product and we find the most similar, value closest to zero, of a pairing where the other product has not already been placed. This results in the retrieval of a product that is somewhat similar to the product we are considering, but is outside of its 8 most similar products. We make this relation clear by connecting the two products with a connection that looks different than every other connection that has been drawn so far. This new product then essentially restarts the process, because it has free products in its CloseProducts which it can place in the free spots around it.

In the end the algorithm should create “islands” of similarity, connected by “bridges”. A diagram of

this can be seen in Figure 8. In each island, the Euclidean distance values between pairings are pretty low, indicating that these products are all more similar. If you take two products from different islands, their Euclidean distance values are going to be higher indicating that they are less similar. This is the best possible result, but in reality, our implementation does not always produce this idealized outcome. Sometimes we have instances of very small islands with only a couple products in them. This likely happens because a product is an outlier and has some nodes in its Closeproducts, but those nodes do not have it in theirs. It also might be that this occurs near the end of the placing where many of the products have already been placed. Likely it is a combination of the two. These are the limitations of our algorithm. There is definitely room for improvement, but we believe this is a good starting point. Ultimately, we have accomplished our goal of creating an interface that sorts by similarity.

4 User Study

This paper claims that the visual interface is more useful to the online shopper than the list interface. To evaluate this claim we ran a user study, where users completed shopping tasks using both types of interfaces. This allowed us to compare the two interfaces and get feedback from the participants concerning any usability problems with our interface.

4.1 Materials

The user study was performed in the Crochet Lab on the Union College campus. The Crochet lab is a controlled environment which is specifically meant for HCI studies. We built the interfaces that the participants would use. To do this we first had to gather product datasets. To gather datasets, we used the Python Beautiful Soup library to scrap product information off of the Best Buy website. We chose Best Buy because it is a large popular retailer that carries many products from various product domains. We scraped four datasets of products to use for the interfaces. We chose four, because we wanted some variety in the products that participants would be exploring in the user study. Each dataset has around 300 products in it. Datasets having close to the same amount of products in them is important because if a dataset had significantly more or less products then the task that we asked the user to do in that dataset might be either easier or harder than the other tasks due to the difference in the amount of things that the participant would have to search through. The chosen product domains were laptops, cameras, televisions, and refrigerators. We then used the data from these four scraped sets to construct 8 different interfaces. Four are of the visual interface type. One visual interface was made for each dataset. The other four were a generic list interface.

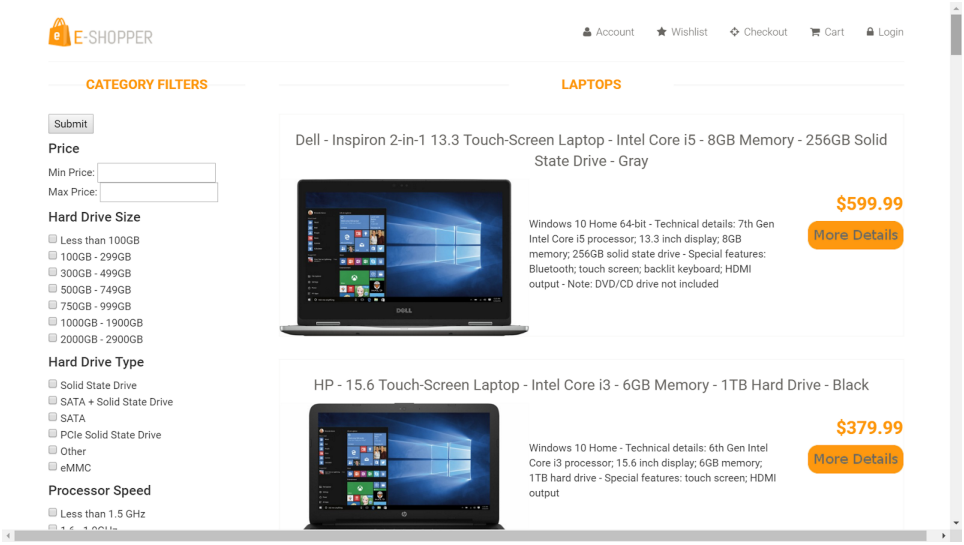


Figure 9: The list interface used in our study is meant to be like any other list interface found on any online shopping website.

An example of the list can be seen in Figure 9. The generic list interface was necessary because we want to compare the behavior of a participant using that to them using the visual interface. Both interface types were implemented with the same look and feel, so as to minimize any nonessential differences that could possibly pose a validity threat. Also the visual interface was built from scratch using JavaScript, while the list interface was obtained as template code for an ecommerce website, and was modified slightly to fit our purposes.

4.2 Methods

We performed a comparative user study to answer the question of whether the interface proposed by this paper is significantly more helpful to the user than the currently used list interface. The participants needed for this study were recruited from the Union College student body. The study itself involved having participants use both interfaces to look for products. Four tasks were given to each participant. Each task has the participant exploring a different product domain using a different interface. The tasks are of the following type: "Please use the interface in front of you to find a laptop that has at least 1Tb of hard drive capacity and at least 10gb of RAM and is something that you would actually like to buy." We found early on that it is very difficult to stimulate genuine shopping behavior in a lab setting. People do not have the actual desire to buy anything and therefore we had two people briefly scan some products and then ended up just picking a product without any real dedication behind their choice. However, we want the user to explore like they would if they were actually shopping using the interface. To remedy this, we slightly tweaked the

task, telling users that they are not shopping for themselves, but rather they are shopping for a gift for a friend. The friend gives some ideas as to the specifications that they want their gift to have. The participants are also gently prompted when they begin that they should explore for a reasonable amount of time before they make their choice. This largely eliminated the previous behavior of participants completing a task in less than a minute.

When we conducted the actual study, a participant swapped which interface they used after completing each task. There are four tasks in total, one for each product domain. If a participant started with the visual interface, then on the next task they would use the list, and so on. Additionally, the tasks were kept in the same order for every participant, however the interface with which the task was completed was swapped between every participant. For example, if one participant was tested for task 1 with interface A, task 2 with interface B, task 3 with interface A and task 4 with interface B, then the next participant would do each task with the opposite interface. This switching would keep happening for every new participant. This process is necessary for the sake of gathering quality data. If we did not switch the interface, then it is a valid criticism of our study that maybe task 1 is inherently more easy than task 2. If we always test task 1 with interface A and task 2 with interface B, and we find out that people complete task 1 faster, then we cannot dismiss that this might be from a difference in the difficulty of the tasks. Swapping the interface used on the task for each participant gets rid of this concern.

The data that was gathered during the course of the study concerned: how fast the users performed a task, what products they explored in the process of completing a task, and feedback from the participants about both interfaces they used. We hoped that the time data representing how fast a user completes a task divided by the amount of products they explored in the process would give us some indication of how helpful one interface is compared with the other. The data on which products users explored allows us to see how people are exploring, and if exploration is significantly different between both interfaces. The feedback about the interfaces is useful for rooting out any usability issues present in our visual interface.

5 Results and Analysis

We expected that we would be able to conclude something about the usefulness of our new interface relative to the list interface. That is, whether it is better or worse for the shopping task at hand. However, a few problems occured that make this comparative analysis impossible. As we noted before, we found out very quickly that it is not feasible to bring a participant into the lab, give them a task to shop for a product, and expect their behavior to be comparable to what it would be if they were shopping on their own. We noticed this because initially people very briefly looked around and then chose whatever product they happened

	Task 1	Task 2	Task 3	Task 4
List	1:03	0:35	0:40	0:53
	1:10	3:20	2:16	4:00
	4:40	13:54	4:24	8:50
	14:25	4:54	4:25	3:16
	12:09	3:14	5:37	4:44
	6:23	4:58	2:56	4:05
Average =	7:45	6:04	3:55	4:59
Visual	0:54	1:10	1:10	0:50
	3:04	3:56	2:15	2:03
	7:46	3:57	4:47	3:10
	4:12	5:39	3:52	3:40
	2:26	7:28	2:15	3:00
	3:05	3:56	4:40	4:57
Average =	4:06	4:59	3:33	3:22

Table 1: Times that participants had for each task, with the list interface on top and the Visual interface on the bottom. Time is measured in seconds.

to be on. This changed a bit when we changed the task to say that they were shopping for a gift for a friend rather than for themselves and told them to use the interface to explore for a reasonable amount of time. However, there is still a problem in that the participants had nothing on the line, so they didn't really have any motivation to truly find the best choice for them. If they were to actually put up their own money and say that at the end of the study, they would actually purchase their choice, we believe that this would have changed their behavior to be more like what it would be naturally. Additionally, all the participants had no background knowledge in at least one of the product domains that they explored for the study. Whereas, if they were at home and were shopping for a gift for a friend, they would be able to research a product and find out about what attributes are good for that product. In this study, they were thrown in the shopping task without necessarily knowing what they should be looking for. That being, what are the best attribute values that a product could have that they should be on the lookout for. Due to the factors of lack of desire, not having their own money on the line and lack of knowledge about products, the data which we collected is open to question.

The numeric results from our study can be seen in Table 1. It is notable that the average value of time for each task is faster with the visual interface. However, we still stand behind the claim that these results are likely not indicative of anything. This is because of the validity problems that were stated before, as well as an insight that prehaps a measure of time is not even a proper metric by which to compare these two interfaces with shopping tasks. We think this because shopping is not really a task that people do for

the sake of speed. The success of shopping is more measured in the satisfaction of the customer in what they have purchased and not in how fast they came to that purchase.

What the user study we conducted truly helped us with was identifying usablilty problems in our new interface. In the interviews with the participants after they completed the tasks, they were asked some questions. Ten out of the twelve participants said that they preferred the list interface to the visual. When asked why they chose the list, one reason that came up many times was that they were more familiar with the list, and therefore they knew the proper way to use it, and were more comfortable with it. Another reason that they gave was that they did not know where to start when presented with the visual interface. For instance, with the list interface there is a very clear indication of what a user is supposed to do. They start their search with the product that is displayed on the top and then they scroll down and look at products as they scroll. The interface of the visual graph does not insist upon an exploration strategy in the same way as the list does. Some participants said that they were overwhelmed when initially presented with it. In the end, participants simply picked a spot on the screen and begin to follow the lines around to different products. It is clear that our interface needs an added mechanism to insist upon a proper exploration strategy for which all users would then use.

The other big problem that participants were having with the visual interface was that they were confused as to what sorting products by similarity actually meant, and how the display in front of them was supposed to convey that to them. When introduced to the visual interface, it was explained to each participant that each product within an "island" was relatively similar and that products from different islands were relatively different. However, even knowing that they did not really get an actual sense of why the things in any island are actually similar. This is a fair criticism. What we meant to convey was that the similarity was based on all of the attributes that the product possesses, as discussed in Section 3.1, however this kind of similarity measure to sort the interface is not exactly intuitive for the user to use in shopping. The sorting is based on every single attribute of the product. This leads to good sorting by Euclidean distance, but it is not very intuitive to be useable by the customer to effecitively shop. Perhaps what we needed to do was lessen the amount of dimensions that we calculated similarity on. Then maybe we could actually indicate on the display what the general attributes and trends are in an island structure.

6 Future Work and Conclusion

Even though we ended up not being able to say anything about whether our type of visual interface is better or worse for shopping than a list interface, we now have useful feedback about our specific implementation. In future work, to deal with the problem of our interface not insisting on a proper exploration strategy, we

could potentially implement an additional mechanism which does. This could be suggestions from each island structure of products to start ones search on. It could even be a list interface off to the side of the visual which suggests products to start on using the search engine algorithm discussed in Section 1. This could be the best of both worlds, utilizing the structure of both types of interfaces to allow the customer to shop better. To deal with the problem of a lack of understanding of what similarity is trying to convey, we could do what we touched on before which is to try to narrow our idea of similarity to the more conventional sense of one dimensional sorting of products based on the value of that single attribute and offer the user the ability to switch the sorting to different attributes. We could also leave the similarity the same and instead, to give more context to why things are similar, we could explicitly state on the connections between nodes what are the attribtues of these products that are the same and which are the ones that are different and by how much are they different. This would work to at least give context when users are exploring as to why these things are placed by each other. With revisions to our interface made, we could conduct the study again. We believe that once the usability problems are dealt with, the best way to compare the list interface and our own is to deploy them on a real online shopping website and let real customers use them. We would be able to see whether the sales volume of the website changes or some other metric, and we would not have to worry about validity threats. In conclusion, we cannot really conclude anything. However, we are hopeful from this study that with the proper implementation our ideas, an interface that sorts by similarity can be an adequate replacement for a common list.

7 References

[1] Doug Bryan and Anatole Gershman. Opportunistic exploration of large consumer product spaces. In Proceedings of the 1st ACM conference on Electronic commerce (EC '99). Pages 41-47. New York, NY, USA, 1999. ACM.

[2] Casner, Stephen, and Jill H. Larkin. Cognitive efficiency considerations for good graphic design. No. AIP-81. CARNEGIE-MELLON UNIV PITTSBURGH PA ARTIFICIAL INTELLIGENCE AND PSYCHOLOGY PROJECT, 1989.

[3] Nish Parikh, and Neel Sundaresan. Marco Polo: A System for Brand-based Shopping and Exploration. Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM '11, pages 2537-2540, New York, NY, USA, 2011. ACM.

[4] Patrick Riehmann, Jens Opolka, and Bernd Froehlich. The product explorer: Decision making with ease. In Proceedings of the International Working Conference on Advanced Visual Interfaces, AVI 12, pages 423432, New York, NY, USA, 2012. ACM.

[5] Tuukka Ruotsalo, Giulio Jacucci, Petri Myllymki, and Samuel Kaski. Interactive intent modeling: information discovery beyond search. Commun. 58, Pages 86-92. 1 (December 2014), 2014. ACM.

[6] Nicolas Sannier, Ben Mathieu, and Sana Baudry. Comparing or Configuring Products: Are We Getting the Right Ones? Proceedings of the Eighth International Workshop on Variability Modelling of Software-Intensive Systems, VaMoS '14, pages 9:1-9:7, New York, NY, USA, 2013. ACM.

[7] Soon Tee Teoh and Kwan-Liu Ma. PaintingClass: interactive construction, visualization and exploration of decision trees. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '03). Pages 667-672. New York, NY, USA, 2003. ACM.