FundExplorer: Supporting the Diversification of Mutual Fund Portfolios Using Context Treemaps

Christoph Csallner*, Marcus Handte†, Othmar Lehmann‡, John Stasko→
College of Computing/GVU Center
Georgia Institute of Technology
Atlanta, GA 30332-0280

Abstract

An equity mutual fund is a financial instrument that invests in a set of stocks. Any two different funds may partially invest in some of the same stocks, thus overlap is common. Portfolio diversification aims at spreading an investment over many different stocks in search of greater returns. Helping people with portfolio diversification is challenging because it requires informing them about both their current portfolio of stocks held through funds and the other stocks in the market not invested in yet. Current stock/fund visualization systems either waste screen real estate or only visualize the user's portfolio and therefore do not show the potentially huge number of stocks available in the market. Distortion applied to treemaps yields both efficient use of screen real estate and visualization of all data points. We have developed a system called FundExplorer that implements a distorted treemap to visualize both the amount of money invested in a person's fund portfolio and the context of remaining market stocks. The FundExplorer system enables people to interactively explore diversification possibilities with their portfolios.

CR Categories: H.5 [Information Systems]: Information interfaces and presentation; J.1 [Computer applications]: Administrative data processing -- Financial

Keywords: information visualization, context, treemap, distortion, query, financial data, stock market, FundExplorer

1 Introduction

Recently, many people have become reacquainted with the fact that a certain amount of risk is associated with investments in the stock market. Diversification is a well-known strategy to limit the risks of financial investments. Diversification aims at investing in a variety of stocks, preferably from different sectors. By diversifying a portfolio, the risk of the investment is distributed, and losses in one stock or market segment (e.g., the technology sector) can potentially be compensated by profits in other market segments. This principle also holds true for investments in equity mutual funds.

*e-mail: christoph.csallner@cc.gatech.edu
†e-mail: marcus.handte@cc.gatech.edu
‡e-mail: othmar.lehmann@cc.gatech.edu
→e-mail: stasko@cc.gatech.edu

Mutual funds are a popular investment choice for private investors, because they allow people to invest money and have it managed by professional fund managers. In this article, we will concentrate on equity mutual funds, a popular form of mutual funds that invests in stocks. Diversification management of a fund portfolio has a number of challenges.

In general, portfolio diversification manages the relation between the portfolio of a person and the context of the whole stock market. It determines what part of the whole stock market the person's overall investment covers, and whether this coverage and distribution is reasonable. We feel that it would be beneficial to support people with an understandable presentation of the diversification of their portfolios so that they can make informed investing decisions.

Equity mutual funds usually invest most of their money in stocks. If a person invests in funds, then he or she is actually investing in the set of stocks that the fund has purchased. So, for the diversification of a fund portfolio, it is in general not enough to buy different funds. For example, although two funds may seem to be different, they can nevertheless invest most of their money in overlapping stocks. Thus, distributing an investment between these funds would not improve portfolio diversification. In order to determine the optimal fund portfolio, the composition of the funds should be considered. One might argue this is not important because the composition of funds changes over time as the management of the funds continuously adjusts the composition to react to market developments. Many funds are relatively stable, however, following a particular investment style and making long-term investments and infrequent trades. Thus, it makes sense to base the portfolio diversification decision on the present composition of the funds.

In this article we present a system that supports investors in the task of portfolio diversification. First, we clarify characteristics of this task, then we survey existing systems that can be used to support the task. Next, we present the concept of the Context Treemap and show how it was used in building our visualization system, the FundExplorer. We conclude with a discussion of our results and we identify possibilities for improvement.

2 The User Task

Fund portfolio diversification consists of two tasks. The first task is to gain an overview of the diversification in the existing portfolio. More specifically, people must understand and assess the stocks and markets held by their funds. Upon gaining this understanding, investors can determine if the resulting diversification is appropriate and sufficient. Once the diversification goals are set, the next task is to modify the existing diversification structure to meet their needs. If a potential new fund addition is known, then investors need to know how this new fund would influence the diversification of their portfolio. Alternatively, a person may be looking for funds that influence a portfolio in a special direction. For example, if someone wants to

increase an investment in the utilities sector, the person will be searching for funds that invest in this sector, while not creating an imbalance in the other sectors.

3 Existing Systems

A variety of tools are available to support investors in the selection of funds for a portfolio. Most tools and methods mainly utilize performance indicators such as past performance, expense ratios, fund ratings, etc., and they often use tables of some sort. The tools usually list the ten largest holdings of a fund and provide pie charts of how the fund's assets are distributed throughout the market sectors. These visualizations give people little support in determining how a new fund fits into an existing fund portfolio, and how it overlaps with funds already owned.

A treemap is an information visualization technique that uses nested rectangles to create a space-filling representation of hierarchical data [Shneiderman 1992]. It has been noted that treemaps are well suited to support decision-making processes in hierarchical structures [Asahi et al. 1995]. The stock market can be viewed as a hierarchical structure, where every stock is part of a sector. Treemaps already have been utilized successfully in the visualization of stock portfolios and markets, e.g., Jungmeister and Turo's [1992] FolioMap prototype and Smartmoney.com's Map of the Market [Wattenberg 1999]. (Actually, Smartmoney.com uses a variation of the original treemap algorithm in which rectangles exhibit more square aspect ratios. Recently, a number of algorithmic variants like this have been introduced [Bederson et al. 2002]). Because of all these desired capabilities, we decided to utilize a treemap in our visualization.

Smartmoney.com also uses treemaps to visualize fund portfolios (at http://www.smartmoney.com/portfoliomap/ as of March 2003). The website uses a treemap to visualize how much money a person has invested in various funds. While this visualization gives a good overview of how the investments are distributed between the funds, it does not support the diversification process very well because it does not show where the money is really invested by the funds, i.e., the individual stocks.

Another straightforward way to visualize individual's fund holdings would be to create a treemap view of the entire market, like Map of the Market, but use color and/or brightness to indicate fund weighting rather than stock performance.

4 Problem Analysis

It is relatively straightforward to use treemaps to visualize how much money a person has invested, by buying funds, in the various stocks of the various sectors of the overall market. The visualization would map invested money *value*(*node*) to screen area *area*(*node*). In a classic treemap[1] the quantity *area*(*node*) is the product of *value*(*node*) and the quotient of the screen real estate *area*(*parent*) of the node's parent node and the money *value*(*parent*) invested into the parent node. The quotient *area*(*node*) / *value*(*node*) given in pixels per dollar is constant throughout the entire tree:

$$area(node) = value(node) * area(parent) / value(parent)$$

Unfortunately, a classic treemap has a limitation that is problematic for communicating portfolio diversity: only nodes that have an associated (positive) value are shown. For *value*(*stock*) = 0 we have *area*(*stock*) = 0. So, with a classic treemap, only the stocks already owned by the investor are visible. Recall that stocks are leaf nodes in a tree with the following levels given in top down order: root, sector, stock. As the treemap displays this hierarchy using a recursive algorithm, it would transitively eliminate sectors as well if the user has not (yet) invested in any of the sector's stocks. Thus, using a classic treemap algorithm as is supports portfolio visualization but does not support the discovery of unknown parts of the market.

Shneiderman [1992] has noted this issue of disappearing nodes. In his working example of visualizing a file system, this issue is not really a drawback if the user is only interested in the top *n* files wasting most space. But if the user cares about small files, this situation also faces our problem.

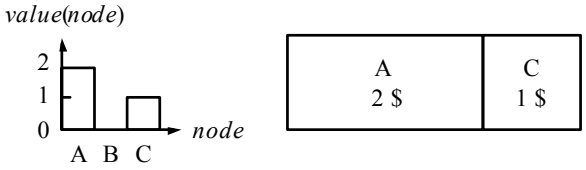

Figure 1: Mapping *value*(*stock*) → *area*(*stock*) visualized by a bar chart on the left and a classic treemap on the right.

Figure 1 illustrates that the usual suspect, "lack of screen real estate", is not the cause of the problem here. The example shows a tiny market consisting of three stocks A, B, and C. A person has invested two dollars in A, zero in B, and one dollar in C. Both portfolio visualizations, the bar chart and the treemap, have enough screen real estate to adequately present the data. Yet the bar chart more clearly shows the values and relationships of *all* data points whereas the classic treemap does not show B. Portfolio diversification is as much about the stocks and sectors not invested in as it is about the stocks and market sectors already invested in. What is needed is a treemap that shows not only the tree nodes (stocks) already present, but also their context.

5 The Context Treemap Solution

Our visualization is based on the notion of distorting the classic treemap visualization, trading proportionality of a mapped attribute for a more inclusive visualization. Figure 2 illustrates the general idea. The distorted treemap (bottom) is a compromise between showing all the data elements—even those with zero value—and a classic treemap that preserves values' proportions (top). So, in many cases a distorted treemap can show one more attribute than a classic treemap, though now the node area is no longer proportional to the attribute visualized. In the following paragraphs we discuss a number of different implementation strategies for this idea.

The first approach uses a regular treemap algorithm, but it modifies the data points' values by applying a function before visualizing them. To work around the problem of non-portfolio stocks having zero value, we can simply add one to the values of all data points. Thus, "absent" stocks receive a value of 1 and all others retain appropriate proportions. Unfortunately, as a portfolio grows in value, the context stocks are essentially squeezed down to be virtually indistinguishable in this strategy.

[1] We use the term "classic treemap" inclusively to refer to any of the existing layout algorithms such as the original technique or the new squarer aspect ratio variants.

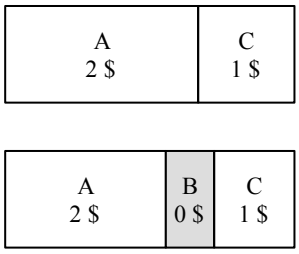

Figure 2: A classic treemap and a Context Treemap displaying the same data set.

An alternative data mapping strategy could use an exponential function to map similar values to more easily distinguishable node areas on screen, as was illustrated by Turo [1994]. For instance, we could use *area*(*node*) = 2^(*value*(*node*)). Such an exponential function maps a zero value to the value one, which is very useful in our case. However, an exponential function approach generally inflates large values while the context (i.e., small values or zero value for us) only receives the remaining fraction of the screen real estate. The fraction approaches zero as the total amount of invested money increases. This holds true even for functions such as 1.0001^(*value*(*node*)), where almost everything is mapped to one until we encounter the explosive exponential growth. Fundamentally, using an exponential function changes the proportionality for non-zero values, which is a serious shortcoming in our problem domain.

Another possible strategy is to assign some minimum screen area to zero value items in order to prevent them from disappearing [Turo and Johnson 1992]. This approach is problematic in that the presence of many zero value items can dominate the display, leaving little room for valued items, and with small-value items essentially disappearing. Furthermore, the strategy requires modifying the underlying treemap algorithm, thus making it less modular.

The solution we developed is an approach that takes ideas from the strategies above. More specifically, it uses another type of distortion function to recalculate node values before display. Conceptually, we reserve a configurable portion of the screen real estate, which is not influenced by the current mix of nodes' values, for the context as follows. First, we calculate the total invested money *value*(*total*). Second, we create an additional amount of money *value*(*total*) * *v* for the context. The screenshots in this article are based on *v* = 0.5. This reserves 33 percent of the total screen real estate for context—stocks in which the viewer has not invested. If there are no investments at all we reserve 100 percent of the space for context. The context screen real estate is split equally among the "empty" stocks by assigning each of them the value $value_c$, where $value_c$ = *value*(*total*) * *v* / *#emptyStocks.* Thus, we utilize the following mapping:

$$value`(node) = \begin{cases} value_c \text{ if } value(node) = 0 \\ value(node) \text{ otherwise} \end{cases}$$

So, for a Context Treemap, we use the following modified chain of functions:
value(*node*) → *value*`(*node*); *value*`(*node*) → *area*(*node*)

The advantages of this approach are that it strikes a good balance between the existing stock portfolio and the context in terms of display real estate, it preserves proportions for held stocks, and no changes to the treemap algorithm are necessary. Thus, it enables the reuse of existing implementations and allows easy switching between them.

6 The FundExplorer

Using the idea of a Context Treemap, we implemented FundExplorer, a tool that supports people in adjusting the diversification of their equity mutual fund portfolios. As shown in Figure 3, a Context Treemap is used to visualize a person's portfolio. FundExplorer shows the stocks in which a person has invested through his or her fund portfolio. The stocks and market segments that an investor has not invested in appear as context.

At first, a person needs to define the initial portfolio he or she wants to visualize. For this task, the FundExplorer provides a list of all available funds in the top left corner of the screen. The user can select owned funds and add them to the portfolio, which is displayed in the bottom of the three regions at the left edge of the screen. Next, the user specifies the amount of money invested in the fund and the color of its representation.

The portfolio shown in Figure 3 consists of one fund. The Context Treemap visualization presents every stock in the market that is available via some fund in the master list. The stocks are grouped by market segment. Each visual item provides a tooltip showing the represented stock's name and sector. Stocks that are held in the investor's funds (the larger rectangles here) are represented by rectangles whose size corresponds to the amount of money invested in them through the funds. The colors of the stock representations are determined by the fund that invests in them—if multiple funds invest in a stock, the stock representation has multiple colors, weighted proportionally. Stocks not held in any of the user's funds are only visible as context and are shown in grey. For the layout of the context treemap, we use a squarified treemap algorithm called pivot-by-size [Bederson et al. 2002] though any of the other algorithms could easily be substituted.

Suppose that an investor already has one fund in the portfolio and wants to diversify this investment. Figure 3 shows that the investor is not invested in the industrial sector—all the stocks in that region are shown as context. So the investor may want to buy additional funds to tap into the industrial sector.

The context feature of the treemap enables investors to utilize the visualization as a query device [Shneiderman 1994]. This capability is perhaps the key feature of the system. In Figure 4 the user has issued a query for funds that invest in the industrial sector by clicking on the border of this sector, which is now marked red. In the center list (marked "Filter") on the left side of the screen, the system then provides a list of funds that invest in the industrial sector. If the investor selects a fund from that list, the FundExplorer shows the stocks of that fund by marking them with blue borders (see Figure 4). Thus, viewers can quickly assess where the new fund overlaps with already owned stocks and how adding this fund would influence the portfolio. By scrolling through the list of funds, an investor can explore which fund may be the best supplement for the existing portfolio. Thereafter, the investor can add the fund to the portfolio and specify the amount to be invested as was described earlier. Individual stocks, as well as sectors, can be used as filtering parameters in this way.

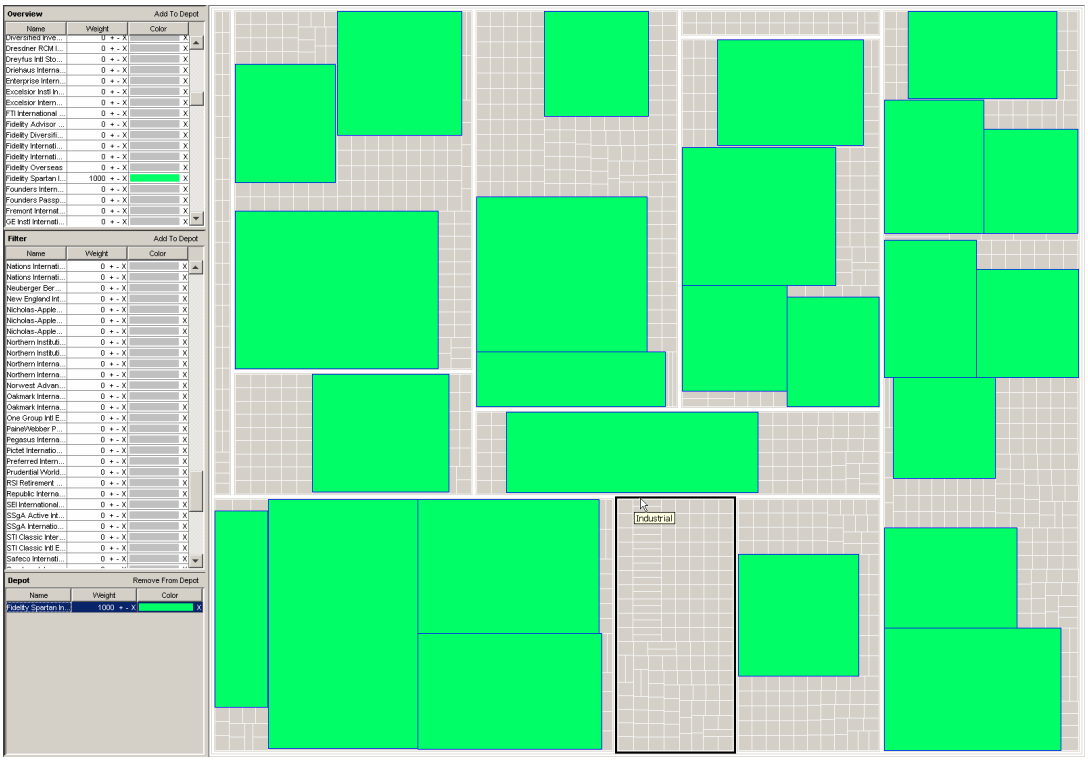

Figure 3: Visualizing one fund with FundExplorer. On the left side, FundExplorer lists from top to bottom: all funds, the funds returned by the current filter and the funds that are part of the depot. On the right side, the context treemap displays the owned fund embedded in its context. Moving the mouse across the context treemap activates tooltips that describe the items pointed at.

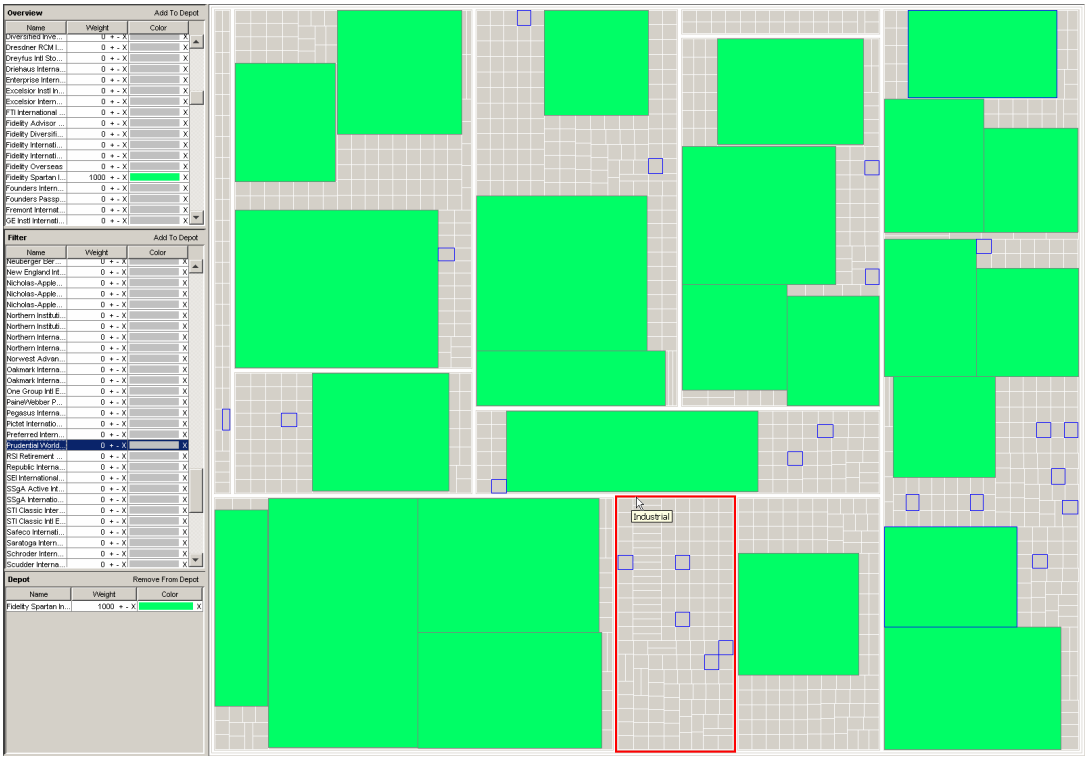

Figure 4: The context can be used to formulate queries. Therefore, all items of the context treemap can be selected and deselected by mouse clicks, and the resulting query is visualized by red borders. Similarly, selecting a fund or set of funds in the overview, filter, or depot view (Prudential is selected in the Filter here) emphasizes the stocks owned by the fund(s) via blue borders.

Figure 5: Stocks held by multiple funds are visualized by rectangles with different colors. The distribution of funds within stocks is simply another level of the visualized tree.

Figure 5 shows the user's portfolio after investing in an additional fund. In this example, the stock in the upper right corner (indicated by the mouse pointer) is common to both funds, so it is rendered in two different colors with the areas of the two regions corresponding to the holding by each fund. Note that any number of funds can be added to a portfolio as desired by the user.

7 Discussion

Enhancing a treemap with context extends its potential applications and uses. The FundExplorer system demonstrates the following two additional applications.

First, the Context Treemap of the FundExplorer enables the visualization of a user's fund portfolio within the stock market. The visualization of data within its context enables an investor to visually determine the diversification of his or her portfolio within the stock market. Additionally, it enables direct comparisons between items (i.e., funds or stocks) that belong to the portfolio and items that are part of the context.

Second, the permanent and complete visualization of the whole stock market enables the formulation of queries. The FundExplorer uses these queries to filter the original set of available funds. The query mechanism provides fast access to funds that have a desired impact on the diversification of an investor's portfolio. The natural formulation of queries provides an easy-to-use interface that also supports iterative selection processes since the user can refine a query visually at anytime.

The combination of these two techniques supports people in optimizing the diversification of their fund portfolios. Compared to the usual table-based approach supported by most existing systems, the FundExplorer enables an easier and faster visual approach based on treemaps. People no longer must create a mental image of their funds within the stock market. They can instead use the treemap representation generated by FundExplorer as a form of external cognition aid. In order to optimize or change their portfolio, people can directly manipulate this representation to achieve the desired effect in terms of diversification.

We have tested FundExplorer with a set of 200 funds that share a common basis of 1000 stocks. Without changes the visualization should easily scale up to 4000 to 5000 stocks on a screen with a resolution of 1024x768 pixels because on average this would leave more than 150 pixels per stock.

Considering that the NASDAQ alone consists of nearly 5000 stocks one might question the applicability of the FundExplorer for the whole stock market. However, we argue that for most tasks of portfolio diversification with funds, it is not necessary to visualize the entire market. First, investors will only be interested in stocks that are owned by at least one fund. If a stock is not part of at least one fund, people do not have a chance to invest in it through funds. Second, investors will be more interested in the stocks that bind a high percentage of a fund's money. Therefore one could reduce the number of stocks by ignoring stocks that fall below a certain threshold. Also, changing the treemap implementation to utilize newer techniques as proposed in [Fekete and Plaisant 2002] could help increase the number of displayable stocks.

Although we are confident that the FundExplorer's Context Treemap is a valuable support tool, the following five aspects of our implementation can be improved.

First, the notion of context generally leads to increased visual clutter. In comparison to a classic treemap in which the size of each node directly corresponds to the value of that item within the tree, the Context Treemap introduces a second meaning for the nodes that belong to the context. The FundExplorer separates those two meanings by using different colors. Such a color scheme does mitigate the problem of different proportions within the same visualization but does not solve it completely. The FundExplorer example shows that the Context Treemap is in general slightly harder to understand than a conventional treemap.

Second, with all the context information presented by the Context Treemap, the performance of the Context Treemap is worse than the performance of a classic treemap when it comes to tasks that do not rely on context information. It is not quite clear whether it would make sense to allow the user to disable the context for tasks that do not rely on context, since switching between two similar representations might also confuse the user.

Another issue that we have discovered during our analysis involves the context itself. The Context Treemap implementation allocates the same weight for all context stocks. Thus, sectors with more stocks have more area devoted to them, which could perhaps be viewed as misleading. In FundExplorer, the context currently serves only as a reminder about stocks not owned, and the user does not perceive any visual clues about the importance of such stocks.

A more elaborate scheme could use domain knowledge to map the domain-specific importance of a node to its size. For instance, while preserving a constant area for the context altogether, the context area devoted to each sector could vary and correspond to the size of that sector relative to the total market. We have not implemented such a mapping to date as it would increase the visual complexity of the tool. More specifically, having different size context nodes in different sectors would complicate the current simple-to-understand mapping. What if a person has

many stocks in a relatively small sector? How big is that sector's area? By trying to map rectangle (stock) area to both current fund holding and to market capitalization, there is greater potential to confuse users. Still, basic functions like specifying queries would benefit from such a custom mapping as the size of a context item would indicate its importance. Clearly, this is a direction to be explored in follow-on work.

Fourth, since the task of diversifying a fund portfolio is usually part of a larger task that aims at balancing the risk-profit-ratio, it is likely that people will be interested in more than just diversification. Other criteria like price-earning-ratio or management cost will be of major interest to the user. Investors will not buy a fund that is likely to produce losses just to achieve better diversification. At the moment, the FundExplorer does not address this issue, that is, it does not visualize additional information within the Context Treemap. Standard GUI techniques like tooltips could be used to present other stock/fund metrics, but this integration likely would not reach the desired level of support for the complex selection process to balance risk and profit. Alternatively, the area, color, and/or other visual attributes such as brightness of an owned stock or a stock in the context could be used to encode any variety of data attributes. Any such addition would increase the visual complexity of the tool, however, making visual inspection and analysis more involved.

Finally, we are presently using the pivot-by-size treemap algorithm that is very "unstable", that is, small changes to data values can cause massive movements of treemap nodes. For the task of exploring alternative portfolios, this is a clear problem. It would be wise to explore alternative layout algorithms such as the strip treemap [Bederson et al. 2002] that are more stable.

8 Future Work

As stated in the previous section, the current implementation of the FundExplorer still has room for improvement. The following ideas could increase the usefulness of the tool and thereby mitigate some of the discussed problems.

First, we could increase the value of the context by weighing its stocks according to different analysis metrics such as, for example, market capitalization. With this enhancement, users would get visual clues about the relevance of stocks that are not part of their portfolio. The knowledge gained from these clues would then support users during the evaluation of their portfolio diversification with respect to market capitalization. The filter mechanism of the FundExplorer would also benefit from this functionality since users will be able to find important stocks much faster. In general, we could enhance the user interface of the system to provide flexible controls for mapping visual attributes, such as area, color and brightness, of both portfolio stocks and context stocks to different stock analysis metrics.

Second, we would like to mitigate the problem of visual clutter introduced by the Context Treemap for tasks that do not necessarily require context. As discussed earlier, one could imagine switching on demand between the conventional and the Context Treemaps, but abrupt changes might be very confusing. A better solution would allow users to specify the percentage of screen real-estate used for the contextual information via an interactor such as a dynamic query slider [Shneiderman 1994]. Smooth transitions would convey the feeling of zooming the context in and out. For this idea to be implemented, a stable (i.e., minimal-jumping) treemap algorithm would be required.

Third, apart from technical optimizations we would like to measure the usefulness of the FundExplorer with representative users. So far we have not performed systematic user studies. People that have seen and used the FundExplorer generally liked the idea and the system, but it would be interesting to compare the table-based selection process and the visual approach using treemaps in a controlled experimental setting. Additionally, it would be interesting to see results of comparisons of the understandability of Context Treemaps in contrast to conventional treemaps as a general visualization.

9 Conclusion

In this article, we have presented a system that makes equity mutual fund portfolios much more transparent and facilitates the task of portfolio diversification. For this system, building on concepts described in [Turo 1994], we have created an extended treemap that visualizes the context of the visualized tree structure. Using the Context Treemap visualization in the FundExplorer system allows us to better show how the investments of a person are situated within the overall stock market and to use the treemap as task specific query device. In summary, our experiences with the FundExplorer are promising, and we believe that the concept of the Context Treemap can be applied to other domains as well.

References

ASAHI, T., TURO, D., AND SHNEIDERMAN, B. 1995. Using treemaps to visualize the analytic hierarchy process. *Information Systems Research 6*, 4, 357–375.

BEDERSON, B.B., SHNEIDERMAN, B., AND WATTENBERG, M. 2002. Ordered and quantum treemaps: making effective use of 2D space to display hierarchies. *ACM Transactions on Graphics 21*, 4, 833–854.

FEKETE, J. AND PLAISANT, C. 2002. Interactive Information Visualization of a Million Items. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis'02)*, 117–124.

JUNGMEISTER, W.-A. AND TURO, D. 1992. Adapting treemaps to stock portfolio visualization. University of Maryland, Center for Automation Research technical report CAR-TR-648 (also CS-TR-2996, SRC-TR-92-120).

SHNEIDERMAN, B. 1992. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics 11*, 1, 92–99.

SHNEIDERMAN, B. 1994. Dynamic Queries for Visual Information Seeking. *IEEE Software*, *11*, 6, 70-77.

TURO, D. AND JOHNSON, B. 1992. Improving the visualization of hierarchies with treemaps: design issues and experimentation. In *Proceedings of IEEE Visualization '91*, 124–131.

TURO, D. 1994. Hierarchical visualization with treemaps: making sense of pro basketball data. In *ACM CHI '94 Conference Companion*, 441–442.

WATTENBERG, M. 1999. Visualizing the stock market. In *ACM CHI 1999 Extended Abstracts*, 188–189.