EUROGRAPHICS - IEEE VGTC Symposium on Visualization (2005)
K. W. Brodlie, D. J. Duke, K. I. Joy (Editors)

# MoireTrees: Visualization and Interaction for Multi-Hierarchical Data

Mahnas Jean Mohammadi-Aragh[†] and T. J. Jankun-Kelly[‡]

Mississippi State University

**Abstract**

*Visualizing hierarchical data is one of the core areas of information visualization. Most of these techniques focus on* single hierarchies*—hierarchies with a single root element and a single path to each element. In contrast, this work focuses on the browsing of* multi-hierarchies*—hierarchies with multiple roots or multiple paths per element. A radial focus+context display algorithm and interaction methods are introduced to explore such multi-hierarchical data. A series of examples demonstrate the effectiveness of our new visualization.*

Keywords: *information visualization, focus+context, radial layout, hierarchical visualization*

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing Algorithms; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction Techniques

## 1. Introduction

Browsing multi-path, hierarchical data is not straightforward due to its interwoven nature. When the hierarchy is loosely defined, searching is difficult without *a priori* knowledge of the underlying organizational structure. Most current database and hierarchical visualization methods do not readily handle loose multi-hierarchies. In this work, we introduce the MoireTree technique, a new focus+context hierarchy visualization, for browsing multi-path hierarchies.

Multi-hierarchical data is common in practice. E-commerce sites often tag products with multiple non-unique categories—the path to a product's information page through different category "parent" pages form a set of overlapping hierarchies. Different taxonomies of the same data form multi-hierachies, as do the union of divergent phylogenetic trees. Though a single, uniform hierarchy could be imposed on this type of data, this will often lengthen or distort the path to leaf nodes. For example, a "uniform" taxonomy would have to introduce new levels in the hierarchy to reconcile branch differences—a portion of a taxonomy under the "Classicial" class in one tree and the "Neo-Classicial" class in other would become a child of a new "Classicial/Neo-Classical" branch in the union of the two trees. Our method alleviates the need for "subset tree" constructions by visualizing and interacting with the multi-hiearchy directly.

There are three main benefits to users of this work. First, the new method allows efficient navigation of interconnected hierarchies through intuitive drill-down and detail-on-demand operations. Secondly, the display of the context of any path into the multi-hierarchy allows users to quickly form mental models of the data. Finally, the focus+context layout increases the amount of effective information navigational displayed. This paper will demonstrate these benefits and its research contributions via a discussion of its relation to previous efforts and several examples.

[†] Visualization, Analysis, and Imaging Laboratory, GeoResources Institute, Engineering Research Center, Mississippi State University, MS, 39762, USA. Email: `jean@gri.msstate.edu`
[‡] Corresponding Author. Department of Computer Science and Engineering, James Worth Bagley College of Engineering, Mississippi State University, MS 39762, USA. Email: `tjk@cse.msstate.edu`

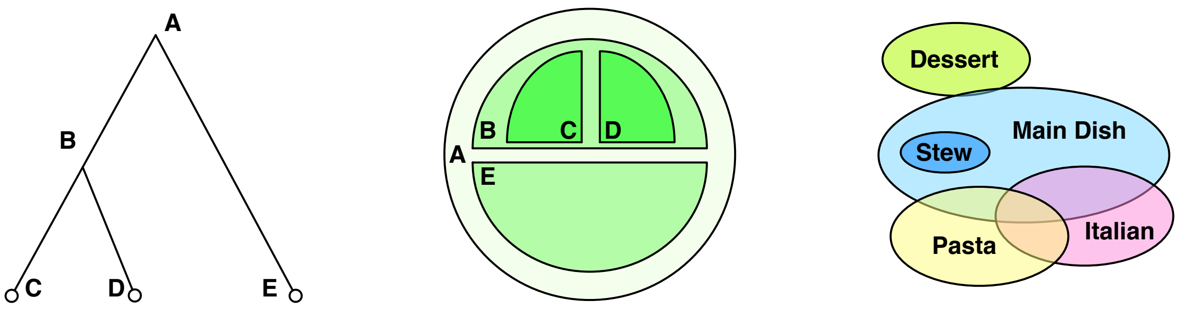


**Figure 1:** *Euler diagram representation (middle) of a traditional hierarchy (left) and an Euler diagram for a multi-hierarchy (right). In a traditional hierarchy, there is one path to any leaf element (e.g., A→B→D); in a multi-hierarchy, there are multiple equivalent paths (e.g., Main Dish→Pasta→Italian vs. Italian→Main Dish→Pasta). Any top level set is a possible root. Equivalent paths are different paths that produce the same results.*

## 2. Related Work

This work draws upon several research areas in information visualization including radial hierarchy visualization, focus+context visualization, and visual database exploration methods. The strongest influences on this work are Zoomology [HDRW03], MoireGraphs [JKM03], and multi-tree hierarchies [FZ94].

### 2.1. Radial Hierarchy Visualization

Radial hierarchy visualizations depict the levels of a hierarchy as concentric circles. Radial space filling hierarchies, such as InterRing [YWRP03], place the root of the hierarchy at the center with each sector of the circles divided among the branches in the tree—a branch with more children would have a larger area than those with smaller children. In contrast, Zoomology [HDRW03] uses a three-level enclosure to display the hierarchy—the outermost circle represents the parent of the current node in the hierarchy while the inner circle contains its children. This use of enclosure is the circular analogy of Treemaps [JS91], which utilizes rectangular regions to densely pack the display space. MoireTrees use an approach similar to Zoomology's, but displays the entire path to the root of the current tree (instead of just the immediate parent) via a concentric display. In addition, the location of children and sibling nodes is constrained to assist the user's mental map of the multi-hierarchy. Combined with a focus+context layout, this allows for a dense information display for browsing.

### 2.2. Focus+Context Visualization

Focus+context visualization attempts to maximize the use of display space by depicting information of interest at a higher resolution/larger size than that of contextual information. For this work, focus+context methods for graph and hierarchy visualization are most relevant. Common approaches use post-process distortion such as "fisheye" lenses [BHDH95, FK95, KR96, SB94, SFM99], projections from a higher dimensional space onto two dimensions [CCF95, KS99], or use non-Eucliean geometry [LR96, Mun98]. Less common are methods that use Euclidean geometry but have distortion effects in the layout itself [JKM03, MGT*03, TM02] or only consider the topology [Noi93]. An in-depth analysis of the different properties of these techniques can be found in Carpendale [CCSF97].

This work adapts our MoireGraph radial focus+context method [JKM03] to radial hierarchy visualization. Our approach can be considered the complement of the MoireGraph—instead of the root of a spanning tree at the center of the display, the center displays the currently selected path element into the multi-hierarchy. Otherwise, the techniques are the same—each subsequent level in the hierarchy is displayed with sequentially less screen space, allowing for a more compact visualization. However, these levels are ancestors in the tree, unlike the MoireGraph (where they are descendants).

### 2.3. Visual Database Exploration and Browsing

Another area of related work is visual database exploration, including browsing. Furnas gives three requirements for effective view navigation: small views, short paths, and logical structure [Fur97]. Also, for strong navigability, the inferred to-set must be the true to-set; users must be able to traverse the visualization without getting confused as to where they are going and for what they are searching. Our method provides navigation cues by rendering the entire path to the current node and context information (alternate branches) at each ancestor level in the multi-hiearchy. As in any hierarchical visualization, any element can be transversed to in $O(\lg n)$, where $n$ is the number of data elements.

Multi-hiearchies are often extracted from databases, where the attributes of the data tables can be extracted to form the hierarchy sets. There are several approaches to database visualization, including parallel coordinates [ID90, War94], scatterplots with brushing and dynamic queries [AS94], pixel-oriented visualizations [KK94], and constructible visualization systems [STH02]. These systems are ideal for finding trends and patterns in databases. In this work, however, the task of interest is browsing, not pattern finding. Though these systems are capable of displaying query results, our approach focuses on the quick construction of the query through direct interaction with the multi-hiearchy. It is conceivable that the two approaches could be combined—a database visualization system could display all the data while a MoireTree could be used to rapidly browse the data.

## 3. Multi-Hierarchies

Our visualization method is targeted at multi-hieachies. A multi-hieachy is a collection of related data elements with multiple hierarchies defined upon it. These hierarchies may be explicit in the data (e.g., there may be known multiple roots of different, overlapping trees) or implicit (e.g., each datum could have distinct but overlapping hierarchies which could be imposed upon them). Multi-tree hierarchies [FZ94] are a form of an explicit multi-hierarchy where an additional constraint of no diamond paths is imposed (i.e., from a single node, there is only one directed path to a descendant). In general, multi-hierchies can possess diamonds in the loosely defined structure.

Conceptually, a multi-hierarchy consists of a collection of nested and possibly overlapping sets. Each set defines a class or attribute of the elements within the set; overlaps occur when elements belong to more than one class. A hierarchy is formed from a collection of nested sets—the root element is the outermost set with each sub-set partitioning the root set into regions of the same class. In a traditional hierarchy, none of the subsets overlap. In a multi-tree hierarchy, overlaps are allowed, but once an overlap occurs, a traditional hierarchy must exist within the overlapping region (this prevents multiple paths). Finally, a multi-hierarchy relaxes this last constraint, allowing arbitrary overlapping sets. Euler diagrams are a traditional way to depict such sets (Figure 1) (Euler diagrams are similar to Venn diagrams, but display set containment instead of relation combinations.).

A concrete example, in this case a recipe database, will clarify the multi-hierarchy concept. The recipe database includes several required attributes, including category, title, ingredients, and directions. Some recipes also include optional attributes such as last preparation date, a ranking, and nutritional information. Most recipes have numerous categories leading to a loosely defined hierarchy. For example, a lasagna recipe can be classified as a member of the "Main Dish", "Italian", and "Pasta" categories; it is unclear which should be considered the "root" category since there are main dishes which are not Italian or pasta, Italian dishes which are neither main dishes or pasta, and so forth. Thus, each category is treated as a root in the multi-hierarchy, and paths to a specific recipe consists of an ordered sequence of category choices such as "Main Dish→Pasta→Italian→Lasagna," "Italian→Pasta→Main Dish→Lasagna," etc. The goal of our system is to provide an efficient means of finding elements within this hierarchy.

Previous efforts to visualize multi-hiearchies impose a uniform hierarchy on the data and then use graph/hierarchy visualization techniques. Graham et al.'s work [GKH00] is notable in that it uses graph and set visualization methods to depict all the hierarchies within a multi-hierarchy divergent taxonomy simultaneously; other efforts have used similar methods to depict portions of multi-hierarchies side-by-side for comparison [HDRW03, MGT*03]. Our approach is in the same vein. However, a MoireTree depicts the multi-hierarchy in-place since it is optimized for browsing instead of comparison.

## 4. Methodologies

The MoireTree system consists of two major components: a multi-hierarchy focus+context layout algorithm and interaction techniques for this layout. The multi-hierarchy is contained within a database; the possible paths formed by the data attributes are generated at runtime. Alternately, the different paths for each item could be enumerated beforehand—this is common in traditional hierarchies.

### 4.1. Layout and Display

A path into a multi-hierarchy induces a tree upon a portion of the multi-hierarchy. A choice on a path forms the intersection of the sets within the path. Each choice is represented as a node in a tree (e.g., a "Main Dish AND Pasta" node). Children of a node correspond to additional possible intersections which exists within the data. Thus, a path induces a node/sub-set duality upon the multi-hierarchy.

Given a multi-hierarchy and a path into the multi-hierarchy, the layout algorithm proceeds over three stages. First, the maximum bounding radius for nodes at each level in the induced tree is determined (Figure 3a). Nodes for subsets/branches within the last element in the current path (the *focus set*) are given a radius $r_{n_0}$ equal to half the display's radius $r_d$. Elements at each subsequent level (i.e., previously within the path) are given half the previous node radius ($r_{n_{i+1}} = \frac{1}{2} r_{n_i}$). From this, the level radii are calculated: given $r_{l_0} = r_{n_0}$, $r_{l_i} = r_{l_{i-1}} + r_{n_{i-i}} + r_{n_i}$. The level radii determine where the node centers are placed for each level.

In the second layout stage, the polar coordinates for each

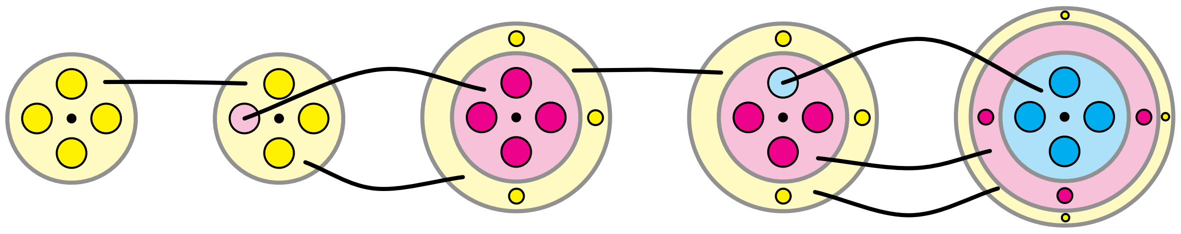
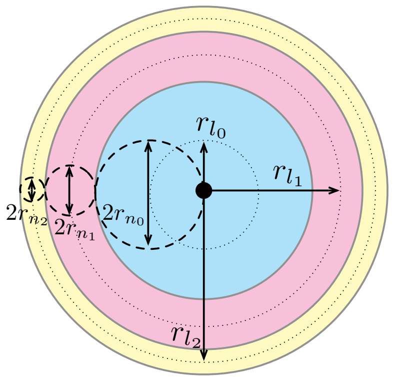
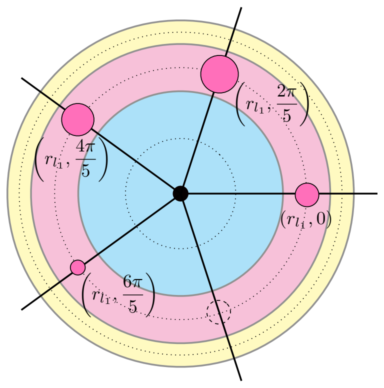


**Figure 2:** *Defining a path in a MoireTree. Starting from the initial display of four sub-sets (left), the user selects the left most sub-set to become the new focus (middle). Finally, the top sub-set in the new focus class is selected, initiating a final transition (right). In the figure, lines connect the same hierarchy level in each display.*



(a) Element and level radii calculation


$\left(r_{l_1}, \frac{2\pi}{5}\right)$
$\left(r_{l_1}, \frac{4\pi}{5}\right)$
$(r_{l_1}, 0)$
$\left(r_{l_1}, \frac{6\pi}{5}\right)$

(b) Polar coordinate calculation

**Figure 3:** *Radial focus+context hierarchy layout. (a) In the first step, maximum node radii $r_{n_i}$ are calculated such that $r_{n_i} = \frac{1}{2} r_{n_{i-1}}$. The level radii (the center of the nodes) are then calculated ($r_{l_i} = r_{l_{i-1}} + r_{n_{i-1}} + r_{n_i}$). (b) In the second step, polar coordinates for each node is calculated. For each node at level i, its radial position is $r_{l_i}$; its angular position is calculated by uniformly dividing the circle for all siblings. A void (the dashed circle) is left for the node that is currently selected at that level. Node size is determined by the number of descendants.*

item in the display is calculated (Figure 3b). The radial position is determined by the level radius. The angular position is determined by dividing the sectors of a circle equally between the number of sub-sets within the level of the hierarchy. In the current implementation, if the number of sub-sets exceed a given number, the remaining sub-sets are not displayed to prevent occlusion. As the sub-sets are prioritized by population, the most relevant choices are always displayed to the user; additional choices (the less populous sub-sets) are displayed in an auxiliary scrolling list for selection.

The final stage of the layout algorithm scales each node's radius by the total number of elements within the node's sub-set—sub-sets with more children are larger than those with fewer children. This radius cannot exceed the maximum node radius for the given level, nor can it be smaller than some specified minimum size. The size differential of the nodes and the ring levels allows the user to easily see what sub-set is focused and which sub-sets are more likely to be relevant to their search.

Once a layout for the current path into the multi-hierarchy has been determined, rendering the hierarchy and the path is straightforward. The nesting category circles are rendered first, with the sub-sets within each set rendered as circles within the level. In the current implementation, categories are treated as nominal variables. Thus a qualitative color scheme using divergent colors was used; this scheme minimizes the chances the user will misinterpret the categories as being related [Bre99]. It is important to note that since we currently use a finite color palette, a deep enough path will cause color cycling. It is believed that the distance between levels with the same color will prevent accidental association between the categories represented by those levels. Text labels for each node and level are included to help identity the portions of the display.

### 4.2. Interaction

The MoireTree system supports a range of interaction methods to facilitate rapid exploration of the data. Interaction methods used in the MoireGraph—focus strength change, display rotation, etc.—are applicable in this radial visualization. Interactions specific to this visualization are query specification (drill-down), undo (roll-up), and obtaining details on demand. The interaction techniques are animated during level transitions (e.g. repositioning of nodes, requerying) to aid in user comprehension; animation uses a slow-in/slow-out method based upon that of Yee et al. [YFDH01].

#### 4.2.1. Defining a query

Upon initialization, the system starts by querying the database for all attributes. The user can then restrict their query in two ways. First, the user can select an attribute from the auxiliary attribute list to restrict the query. This selection will generate a new path into the hierarchy, and the display will be updated accordingly. The user can add restrictions anytime. The list display is static. It does not reflect actual combinations of categories within the data, but all possible categories. Thus, it is possible to create a query for sets which are empty. Sometimes this is a desirable operation, but it is not efficient.

The other method to develop a query is performed by interacting with the node that represents the sub-set directly (Figure 2). As with the first method, this action restricts the query to elements within the intersection of all the currently selected attributes and an animated transition occurs. The user can continue to restrict the query by selecting subnodes. To maintain the user's mental model of the hierarchy, a void in the parent level layout is left where the new focus set previously was; this keeps the position of the set's siblings fixed in the new layout. Thus, there is no unexpected "cross-over" of nodes during the animated transition.

To better illustrate browsing, consider the recipe example. By selecting "Chicken" in the category list, the user restricts the system to only return recipes that were categorized as chicken. The displayed nodes would consist of the top categories within the chicken category. If the user then selected the "Main Dishes" node, an intersection operation would be performed on the Main Dish set and the Chicken set, and the new focus nodes would consist of the top categories within this new tree. This process is repeated as the user selects on more sub-nodes.

#### 4.2.2. Undo

Since a user can continually restrict the query, it is vital they be able to undo a restriction. By clicking the background, the query is unrestricted—the last element in the current path is removed. The user can also select an outer context ring and undo several steps at once. If a sub-node of a context ring (a sub-set in a previous category in the path) is selected, that sub-set becomes the focus; this allows more efficient navigation than just parent/child traversal. An undo function is required since much of exploring is trial-and-error; this method supports rapid evaluation of "what-if" scenarios.

### 4.3. Details on Demand

Once the user has created an exploration path, a method to retrieve complete information on a specific data item is provided. Selecting a leaf node displays the details of the corresponding data element in another window. For example, to print a recipe, the user would first restrict the query and find a recipe or a set of recipes. Then, by selecting the individual recipes, the recipe information (e.g. ingredients and instructions) is returned.

## 5. Examples

We implemented a prototype visualization tool, which we applied to two different multi-hierarchical datasets. The first is a recipe collection consisting of 245 recipes with 134 unique categories. Each recipe has unlimited multiple categories. The second is a paper database consisting of 614 papers with 80 unique keywords. Each paper has up to three keywords.

The number of nodes per level (i.e., number of exploration paths) is variable. To ensure readability in printed form, a maximum of six sub-sets were displayed per level. Increasing the number of displayed sub-sets results in a higher chance for node occlusion (if there are many nodes); in this case, highlighting nodes can be used to remove occlusions.

### 5.1. Recipes by Category

While we found no specific examples of visualizing a recipe collection, any database visualization method could be used to create a recipe visualization. In that case, the resulting visualization would find trends and patterns within the database. In our case, we are interested in browsing the database and finding individual recipes based on appropriate categories. The data is multi-hierarchical since any one of a recipe's multiple categories could be used as a starting point (root node) for browsing. Also, due to the unlimited number of categories per recipe, multiple paths to a single recipe exists. Figures 4 and 5 illustrate this concept.

### 5.2. Papers by Keyword

The 2004 IEEE Symposium on Information Visualization Contest focused on visualizing a paper database (http://www.cs.umd.edu/hcil/InfovisRepository/contest-2004/index.shtml). Most contest entries provided static overviews and visualizations of research areas and author relationships. The database includes keywords (the multi-hierarchy sets we chose), authors, titles, and other meta-data. While many entries focused on providing a summary static view of the database, White et al. [WLB04]

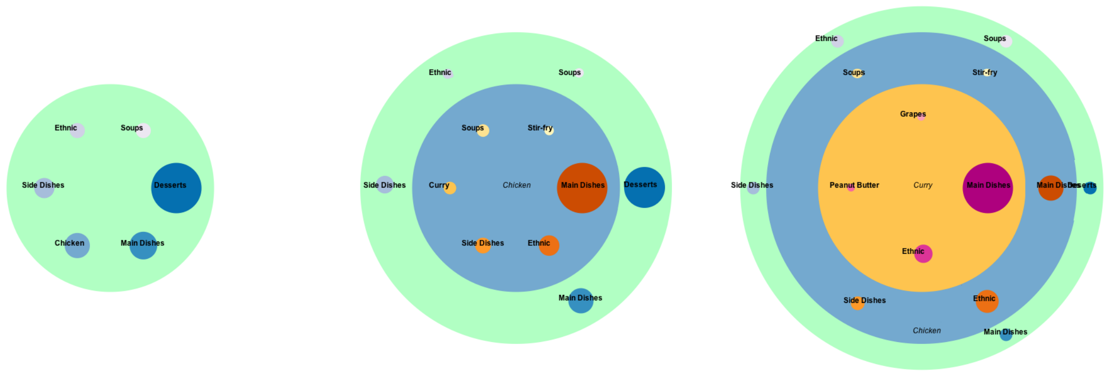


**Figure 4:** *Browsing a recipe collection with a MoireTree. Starting from the initial display of six sub-sets (left), the user selects the "Chicken" sub-set to become the new focus (middle). Next, the "Curry" sub-set is selected and becomes the focus (right). All the node and label transitions are animated.*

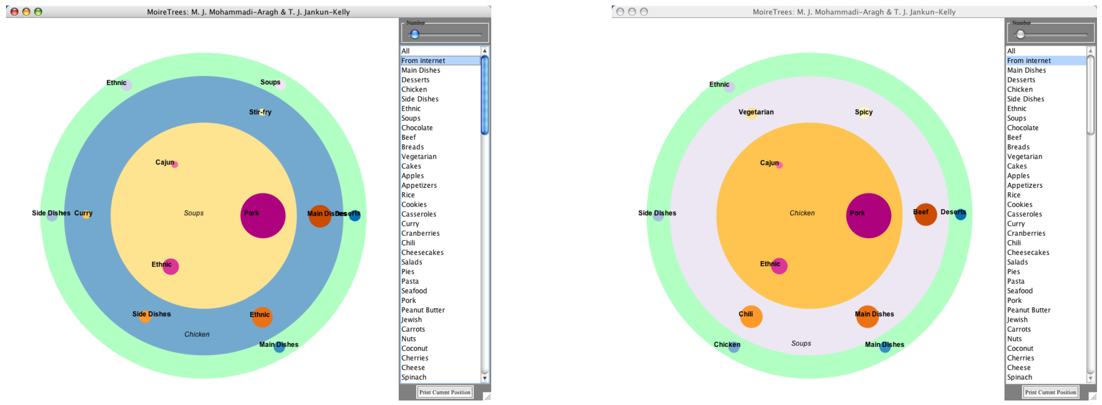


**Figure 5:** *Different views of multi-hierarchical data using MoireTree. To find an internet chicken soup recipe, the user first selects "From Internet" from the category list. Two paths exist. The user can either select "Chicken" and then "Soups" (left), or "Soups" and then "Chicken". Notice that both paths bring the user to the same results.*

is notable in that they provide interactive searching of the keyword multi-hierarchy. However, this form of search regenerates the visual display for each search, potentially confusing a user's mental map of the data. Since the MoireTree displays the current path at each point of the browsing process and animates transitions, the user's mental map is preserved. Figure 6 illustrates browsing the paper database with MoireTrees using keywords. Since papers are limited to three keywords, leaf nodes are reached after three successive queries. Alternatively, a multi-hierarchy could be extracted from author names, highlighting collaborations within the data.

## 6. Current Implementation and Performance

The current MoireTree system is implemented in Java without any hardware accelerated rendering. On a 1.5 Ghz PowerPC G4 with 1GB DDR SDRAM, preprocessing the recipe data took 3.17 s. Preprocessing the paper data took 3.84 s. The layout calculation is dependent on the number of nodes per level. In our case, it took an average of 0.56 s for the recipes and 0.61 s for the papers. Transitions between levels take an additional 2.5 s to animate.

## 7. Preliminary Evaluation

To evaluate the use of multi-hierarchies, an alternate implementation was explored. In this implementation, a strict hierarchical structure for the system was developed, and exploration was restricted to this structure. Using the recipe example, the top-level categories were Main Dishes, Side Dishes, Desserts, Drinks, and Appetizers. Under "Main Dishes", the next level included Meat, Casseroles, Pasta, Slow Cooker, Sandwiches, Soups, Vegetarian, Grilling, and Ethnic. Other categories were filled out appropriately.

Although a formal user study has not been completed,

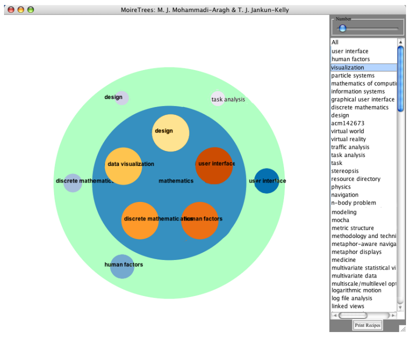


**Figure 6:** *Browsing a paper database with a MoireTree. In the database, papers are limited to three keywords. Therefore, after three subsequent queries, the results are all leaf nodes. The leaf nodes are labeled with the first few words of the title. Clicking on a leaf node prints details of the corresponding data element (e.g., paper title, abstract).*

preliminary user feedback was collected from three users regarding their preferences for the two different methods. Users were asked to describe their process aloud during the exploration and were surveyed after the tests. All of the users preferred the multi-hierarchy version of the database to the strict hierarchy. The former version does not pre-define categories or category order of appearance, simplifying the process of browsing. For example, if a user wanted to look at all beef recipes, the second implementation did not accommodate; the user was always first presented with “Main Dishes”, “Sides”, “Desserts”, “Appetizers”, and “Beverages”. Since our method displays all possible categories (sets) within the multi-hierarchy initially, users were able to select any top category to begin browsing. More importantly, using the multi-hiearchy directly allows the data to define the visualization rather than a programmer’s pre-defined hierarchy. Without personal knowledge of the database, users were able to easily explore the data, form a mental model, and find recipes that met their needs. For other applications where the hierarchy is better defined, a multi-hierarchy would not be as beneficial. However, our visualization method would still provide browsing benefits.

## 8. Conclusions and Future Work

Many datasets possess hierarchies which overlap or are only loosely defined. This work introduces a conceptual model for such data—multi-hierarchies—and a focus+context radial visualization technique to browse such data efficiently. Multi-hierarchies encapsulate a wide-range of data of interest, and the MoireTree algorithm can be adapted to traditional hierarchies to provide the benefits of efficient browsing. Preliminary tests support the conclusion that multi-hierarchies are a natural way to explore data with multiple, related attributes.

When considering if MoireTrees are appropriate for a given application, there are two important factors: the total number of classes/categories overall and the maximum number of classes/categories per datum. The former determines the maximum number of nodes per level (limited by a user set maximum) while the latter determines the maximum number of levels. These two factors control the behavior of the layout. For example, unlimited categories per datum have deep multi-hierarchies (e.g., the recipe database) whereas limited categories per datum have shallow trees (e.g., the paper database). The MoireTree system appears best suited when extensive overlap occurs through a combination of a high total category count and numerous categories per element. In other words, MoireTrees provide the most benefit when numerous equivalent paths to an element or group of elements exist. When there is little or no overlap, a database or strict hierarchy visualization system may more efficiently use screen space and provide adequate browsing capabilities.

In the future, further testing of the system is desired. The ultimate scalability of this approach is an open question, and alternate display methods for multi-hierarchies are desired in order to evaluate these limits. One area of further investigation is further compaction of the node layout. To better utilize the increased perimeter as one moves away from the focus, node placement could be relaxed to either allow for the overlap of nodes or allow for nodes not strictly alined with the level radius. Both of these design choices have consequences (i.e., occlusion or loss of context) that need further study. In addition, a method to scale node size based upon the number of siblings should be investigated to reduce occlusion. Formal user studies would also quantitatively measure the cognitive benefits and trade-offs of using multi-hierarchies vs. traditional hierarchies.

**Acknowledgments**

The authors thank the members of the Mississippi State ERC Visualization, Imaging, and Analysis Laboratory for their comments and discussion. We also thank Kwan-Liu Ma and Soon Tee Teoh for their involvement in the development of the MoireGraph, the inspiration for this work.

## References

[AS94] AHLBERG C., SHNEIDERMAN B.: Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *Proc. of the SIGCHI conference on Human factors in computing systems* (1994), ACM Press, pp. 313–317. 3

[BHDH95] BARTRAM L., HO A., DILL J., HENIGMAN F.: The continuous zoom: A constrained fisheye technique for viewing and navigating large information spaces. In *Proc. of the ACM Symposium on User Interface Software* *and Technology* (1995), Information Navigation, pp. 207–215. 2

[Bre99] BREWER C. A.: Color use guidelines for data representation. In *Proc. of the Section on Statistical Graphics* (1999), American Statistical Association, pp. 50–60. 4

[CCF95] CARPENDALE M. S. T., COWPERTHWAITE D. J., FRACCHIA F. D.: 3-dimensional pliable surfaces: For the effective presentation of visual information. In *Proc. of the ACM Symposium on User Interface Software and Technology* (1995), Information Navigation, pp. 217–226. 2

[CCSF97] CARPENDALE M. S. T., COWPERTHWAITE D. J., STOREY M.-A. D., FRACCHIA F. D.: *Exploring Distinct Aspects of the Distortion Viewing Paradigm*. Tech. Rep. 97-02, School of Computing Science, Simon Fraser University, 1997. 2

[FK95] FORMELLA A., KELLER J.: Generalized fisheye views of graphs. In *Proc. 3rd Int. Symp. Graph Drawing, GD* (Berlin, Germany, 1995), Brandenburg F. J., (Ed.), no. 1027, Springer-Verlag, pp. 242–253. 2

[Fur97] FURNAS G. W.: Effective view navigation. In *Proc. of the SIGCHI conference on Human factors in computing systems* (1997), ACM Press, pp. 367–374. 2

[FZ94] FURNAS G. W., ZACKS J.: Multitrees: enriching and reusing hierarchical structure. In *Proc. of the SIGCHI conference on Human factors in computing systems* (1994), ACM Press, pp. 330–336. 2, 3

[GKH00] GRAHAM M., KENNEDY J. B., HAND C.: A comparison of set-based and graph-based visualisations of overlapping classification hierarchies. In *Proc. of the working conference on Advanced visual interfaces* (2000), ACM Press, pp. 41–50. 3

[HDRW03] HONG J. Y., D'ANDRIES J., RICHMAN M., WESTFALL M.: Zoomology: Comparing two large hierarchical trees. In *2003 IEEE Symposium on Information Visualization Poster Compendium* (unpublished), 2003. 2, 3

[ID90] INSELBERG A., DIMSDALE B.: Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Proc. of the 1st conference on Visualization '90* (1990), IEEE Computer Society Press, pp. 361–378. 3

[JKM03] JANKUN-KELLY T. J., MA K.-L.: Moiregraphs: Radial focus+context visualization and interaction for graphs with visual nodes. In *Proc. of the 2003 IEEE Symposium on Information Visualization* (2003), IEEE Computer Society Press, pp. 59–66. 2

[JS91] JOHNSON B., SHNEIDERMAN B.: Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proc. of the 2nd conference on Visualization '91* (1991), IEEE Computer Society Press, pp. 284–291. 2

[KK94] KEIM D. A., KRIGEL H.-P.: VisDB: Database exploration using multidimensional visualization. *IEEE Comput. Graph. Appl. 14*, 5 (1994), 40–49. 3

[KR96] KEAHEY T., ROBERTSON E.: Techniques for non-linear magnification transformations. In *Proc. of the IEEE Symposium on Information Visualization '96* (1996), pp. 38–45. 2

[KS99] KREUSELER M., SCHUMANN H.: Information visualization using a new focus+context technique in combination with dynamic clustering of information space. In *Workshop on New Paradigms in Information Visualization and Manipulation* (1999), pp. 1–5. 2

[LR96] LAMPING J., RAO R.: The hyperbolic browser: A focus+context technique for visualizing large hierarchies. *Journal of Visual Languages and Computing 7*, 1 (1996), 33–55. 2

[MGT*03] MUNZNER T., GUIMBRETIÈRE F., TASIRAN S., ZHANG L., ZHOU Y.: TreeJuxtaposer: Scalable tree comparison using focus+conext with guaranteed visibility. *ACM Transactions on Graphics 22*, 3 (2003). 2, 3

[Mun98] MUNZNER T.: Exploring large graphs in 3D hyperbolic space. *IEEE Computer Graphics and Applications 18*, 4 (/1998), 18–23. 2

[Noi93] NOIK E. G.: Layout-independent fisheye views of nested graphs. In *Proc. for the 1993 IEEE Symposium on Visual Languages* (1993), pp. 336–341. 2

[SB94] SARKAR M., BROWN M. H.: Graphical fisheye views. *Communications of the ACM 37*, 12 (1994), 73–84. 2

[SFM99] STOREY M.-A. D., FRACCHIA F. D., MULLER H. A.: Customizing a fisheye view algorithm to preserve the mental map. *Journal of Visual Languages and Computing 10*, 3 (1999), 245–267. 2

[STH02] STOLTE C., TANG D., HANRAHAN P.: Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics 8*, 1 (2002), 52–65. 3

[TM02] TEOH S. T., MA K.-L.: Rings: A technique for visualization of large hierarchies. In *Proc. of Graph Drawing 2002* (4 2002), Goodrich M., Kobourov S., (Eds.), Springer-Verlag. 2

[War94] WARD M. O.: Xmdvtool: integrating multiple methods for visualizing multivariate data. In *Proc. of the 5th IEEE Conference on Visualization (Vis' 94)* (1994), IEEE Computer Society Press, pp. 326–333. 3

[WLB04] WHITE H., LIN X., BUZYDLOWSKI J.: An associative information visualizer. In *2004 IEEE Symposium on Information Visualization Poster Compendium* (unpublished), 2004. 5

[YFDH01] YEE K.-P., FISHER D., DHAMIJA R., HEARST M.: Animated exploration of dynamic graphs with radial layout. In *Proc. of the 2001 IEEE Symposium on Information Visualization* (2001), IEEE Computer Society, pp. 43–50. 5

[YWRP03] YANG J., WARD M. O., RUNDENSTEINER E. A., PATRO A.: Interring: a visual interface for navigating and manipulating hierarchies. *Information Visualization 2*, 1 (2003), 16–30. 2