J.T. Yao, V.V. Raghvan, G.Y. Wang (Eds.): WSS'04, pp. 67-74, 2004.

An Integrated Approach to Discovery in Complex Information Spaces

Daryl H. Hepting and Cory Butz

Department of Computer Science, University of Regina
Regina, Saskatchewan, S4S 0A2, Canada
E-mail: {dhh,butz}@cs.uregina.ca

Abstract

As the amount of available data continues to increase, more and more effective means for discovering important patterns and relationships within that data are required. Although the power of automated tools continues to increase, we contend that greater gains can be achieved by coordinating results from a variety of tools and by enhancing the user's ability to direct the application of these tools. A system which can rely on multiple modalities for processing information has a distinct benefit in terms of user-confidence in the final results. We set forth an approach which permits a flexible, user-controllable model of the information space within which basic tools can be integrated. The analysis of data, whether it be through visualization or data mining, for example, is an exercise in problem-solving and any computer-based tool to support the analysis process should be designed to support problem-solving activities. The process by which a user can develop and interact with this model is described and the benefits of this approach are discussed. This integration can be extremely useful both for the development of new hypotheses regarding the data and for verification of existing hypotheses.

1. Introduction

Visualization in scientific computing continues to gain prominence as a tool for data analysis and comprehension, beginning with the landmark report of McCormick *et al.* [20]. In the modern era, this trend can trace its roots back to the beginnings of modern computing. Even before the advent of computer graphics, numerical simulations were an important tool for scientific insight. In the 1940's, John von Neumann [37] wrote about the potential for the use of computers in the study of differential equations. It is this potential of the computer as an experimental tool which caused Richard Hamming [9] to write "the purpose of computing is insight, not numbers" as the motto for his 1962 text, *Numerical Methods for Scientists and Engineers*. It is important to remember that insight is the goal of any computer-aided analysis activity, whether it be scientific visualization, data mining, or machine learning.

Although Jessup [12] contended that scientific visualization has the promise to democratize visual thinking, the capability to produce computer-generated visual representations alone is insufficient to realize this promise of aiding the achievement of insight for individuals. What is true for visualization is also true in a much more general sense for other forms of computer-based analyses. The mere existence of a capability is not sufficient to have it adopted and used successfully by all those who might gain insight with it. In general, tools must be created that allow access to the method without requiring the user to be an expert in the vocabulary associated with the details of the method. More generally though, tools should present the user with a representation of the context in which the method is invoked. Furthermore, these tools should enable the domain expert to work with the analysis tools, without being burdened by the need to learn a specialized vocabulary, to produce representations which are effective for the expert [32]. In the realms of scientific and information visualization, the *cogito* system has been a example of this paradigm [10].

Consider that any visual representation can be decomposed into *components*, each with their own *elements*. A component could be "graph type", with elements including "bar chart", "pie chart", "line chart", "scatterplot", and so on. Each visual representation can be denoted as an N-tuple, where e_i is an element of component C_i. In practice, not all N-tuples will correspond to valid visual representations because of incompatibilities between elements of different components. The Cartesian product of the elements from all the components forms the N-dimensional space of available visual representations

$$\langle e_1, e_2, \ldots, e_N \rangle \in C_1 \times C_2 \times \ldots \times C_N.$$

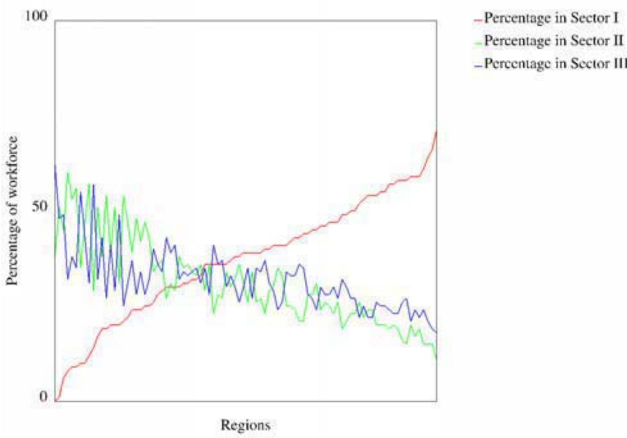

Figure 1: One possible visual representation of Bertin's data, constructed from components and elements that specify features including the graph type, the annotation, the sorting of the data, and the colours.

The space of available visual representations can be very large and it can be difficult to grasp the implications of all available combinations of elements. It is particularly clear in this type of situation that it is not possible to completely articulate all elements of a problem, *a priori*, to arrive at a solution. This fact only exacerbates the problem of selecting and specifying the individual elements in a visual representation. In the midst of so many combinations, it can be difficult to find a visual representation which is apposite. As Wegner [38] points out, interactive systems can offer a great deal more power in dealing with this situations, over algorithms alone. Researchers have observed that problems and solutions coevolve [6]. One usually does not, perhaps cannot, have a clear statement of the problem without an idea of the solution. Rittel [25] described this as follows: "you cannot understand the problem without having a concept of the solution in mind; and that you cannot gather information meaningfully unless you have understood the problem but that you cannot understand the problem without information about it." One must simply begin. The evolutionary nature of the design process is well-described by the model of evolutionary epistemology [31]. Allen [1] uses this formulation, based on Neill [21], for information seeking.

The *cogito* system supports each user in this process by providing external representations of the space of available alternative combinations and the means to manipulate it quickly and easily, through a predominantly visual interface [11]. Although the *cogito* system was developed as a means to support the scientific visualization process, the principles are amenable to other areas. The recent work by Grady *et al.* [8], which illuminates several important open problems with respect to the integration of data mining and visualization processes, echos key points of the design philosophy in *cogito*. Thus, we feel that our system coincides nicely with the architecture of web-based support systems suggested by Yao and Yao [44].

The rest of this paper is organized as follows. Section 2 describes the paradigm embodied by the *cogito* system. Section 3 describes methods for learning which can be incorporated within the *cogito* framework for discovery. Section 4 describes how the automated capabilities can be integrated within *cogito*. Finally, Section 5 presents some conclusions and directions for future work.

2. Design of the *cogito* system

The use of components and elements to describe particular visual representations is an adaptation of Bertin's [2] retinal variables which he used to systematically explore marks on a plane and how those marks could be used to construct diagrams, networks, and maps. Graphic communication in two dimensions has been thoroughly studied and the construction of visual representations within this realm is fairly well understood. For this reason, the example chosen to illustrate this presentation and to evaluate the prototype software is a two-dimensional graphing problem based on a small dataset from Bertin [2][page 100]. It provides a view of the French economy from the early 1960's. For each département in France, the data provides the workforce (in thousands of workers) for each of the three sectors (primary, secondary, and tertiary) in the economy; the total workforce (the sum of the three sectors); and the percentage of the workforce in each sector. Figure 1 presents a sample visual representation of this data.

Bertin [2] remarked that "to construct 100 DIFFERENT FIGURES from the same information requires less imagination than patience. However, certain choices become compelling due to their greater 'efficiency."' But the question of efficiency is closely linked to the task at hand and the user's experience with the elements of a visual representation, as Casner [3] describes. Although Bertin contends that meaning can be communicated fully through a graphic and its legend, the more widely accepted view is that communication and interpretation occur, or are influenced by things, outside this realm. For Winograd and Flores [39], this means that "the ideal of an objectively knowledgeable expert must be replaced with a recognition of the importance of background. This can lead to the design of tools that facilitate a dialog of evolving understanding among a knowledgeable community." This type of computer-based tool can be hard to construct.

Adapting the classification of Kochhar *et al.* [14], it is possible to distinguish manual, automatic, and augmented systems based on their relationship of human and computer.

Manual systems require the user to completely describe and control the operation of the visualization application. The space of alternatives available for exploration in these schemes is implicitly limited by the user's own experience. Systems exemplified apE (animation production Environment) [5] and AVS (Application Visualization System) [35], are collectively known as Modular Visualization Environments (MVE's). MVE's have come to prominence because they allow users to create complete visualizations from components connected using a visual dataflow model. DataDesk, the statistical graphics package first described by Velleman and Pratt [36] in 1989, provides a direct-manipulation interface to statistics and a good example of Tukey's Exploratory Data Analysis [34]. It builds on the idea that multiple, connected views of data can greatly enhance the power of data analysis. Graphical interfaces are seen as ways to specify "like this one, only different in the following ways." Insight is acknowledged as important. The Spreadsheet for Information Visualization (SIV) [4], based on work presented by Levoy [17], is a novel use of the spreadsheet programming paradigm that allows the user to explore the effect of the same operation on several related images.

Automated systems appear to the user as black boxes which are given input and produce output. The rationale behind them is that the number of alternative visual representations is so large that the user would be overwhelmed if he or she had to deal with the space in its entirety. In accepting this guidance from the computer, the user relies more on the computer for its application of design rules and gives up more freedom to exercise personal choices about what the visual representations will contain. In 1986, APT (A Presentation Tool) by Mackinlay [18] contributed a formalization of the design rules for two-dimensional static graphs, based on Bertin [2] and others. It was a prescriptive system because it chose graphics on the basis of expressiveness and effectiveness criteria. With BOZ in 1991, Casner [3] added information about the task to his presentation system and this resulted in a noticeable improvement in user performance with the graphs that his system generated.

Augmented systems aid the user by allowing certain well-defined tasks to be performed primarily by the computer, with the effect of increasing the capabilities of people to tackle complex problems. Because any articulation of a design is an ongoing process which is necessarily incomplete, it is important for the user to maintain some control. Rogowitz and Treinish [26] described a visualization architecture that allowed the user to choose a higher-level interaction with the visualization process, based on the invocation of appropriate rules. The VISTA (VISualization Tool Assistant) environment described by Senay and Ignatius [29] would analyse, as much as possible, the input data and suggest a visual representation to which the user could make modifications. The SageTools [27] system allowed users to work in the context of past graphics with the option to modify what had already been done. The Integrated Visualization Environment (IVE) [13] implemented the cooperative computer-aided design (CCAD) paradigm. It used a generative approach, in which the user could intercede after each iteration to select promising designs for further development. Design Galleries [19] worked to provide a good sampling of the range of alternatives. The user specified the means for comparison and the system worked offline to generate and evaluate the images based on the user's specification and then displayed the results.

Rather than focus on the results produced by these visualization systems and attempt to answer whether a "best" visual representation can be decided for any context or any group of users, it is productive to look at the process by which these representations can be developed. According to Winograd and Flores [39], we can "create computer systems whose use leads to better domains of interpretation. The machine can convey a kind of coaching in which new possibilities for interpretation and action emerge."

Norman [22] describes the twin gulfs of execution and evaluation. With a goal in mind, a user experiences the gulf of execution in deciding which commands to execute in order to move from his or her present state to the goal state. Similarly, the gulf of evaluation is encountered when a user tries to reconcile an intermediate result state with the original goal state. An effective interface will minimize these gulfs, and for visualization tasks a visual interface is indicated.

In 1991, Sims [30] presented a method for the use of artificial evolution in computer graphics which employed both a genetic algorithm [7] and genetic programming [15]. Both of these "genetic" methods work by simulating the cellular-level processes of cross-over and mutation. The former does this as means to search a space whereas the latter works to transform the space itself. For Sims, the goal was to evolve images and textures. However, because it can be surprising to see images from different generations with no apparent connection between them, it can work to defeat the user's control. In 1992, Todd and Latham [33] also discussed a genetic approach to the generation of new images, theirs being more restrictive and controllable by not including genetic programming.

Even for small problems with relatively few alternatives, an exhaustive evaluation is almost always completely impractical. Instead, humans rely on heuristic search methods which are likely to find acceptable solutions in a reasonable amount of time. These search heuristics can be of two sorts, in general. If the problem is well-understood, local search techniques may be employed effectively. If the problem is new, a global search may be better suited to the exploration of alternatives.

The *cogito* software system was designed to address the shortcomings of traditional visualization tools. In particular, the system deals with the problem of articulation with a visual interface that provides non-verbal access to alternatives. With an incomplete articulation of the context, the iteration performed in selecting and evaluating candidate visual representations is crucial to the visualization process. The evaluation of visual representations can be done more effectively if the available alternatives are understood, and interaction is essential to accomplish this. The *cogito* system supports "combinatory play" by considering every visual representation to be the product of elements from each of several components and it relies on the user to choose these elements. Not only is this conception of components and elements familiar from Bertin, it also occurs in MVE systems like AVS [35] (Application Visualization System), and the toolkit philosophy of the Visualization ToolKit [28]. But, in *cogito*, the user does not choose these elements in isolation. Rather, he or she chooses between whole visual representations, each of which comprise particular elements.

The computer is well-suited to provide such external memory to support this decision-making process. Placed between manual and automatic systems, the design of *cogito* uses the computer to perform bookkeeping functions and allows the user evaluate and select. A traditional visualization system, with its need for expertise in programming, can separate the user from this important function. Whereas programming support is also required for *cogito*, the user and the programmer may work together to create the notion of the space of available representations and the user is still able to interact directly with the computer. Figure 2 illustrates this difference.

The *cogito* system provides, through views, the means to structure and examine the space according to a range of criteria. The user sees the current space, with the current organizational view, one screen at a time. Cells, which display individual visual representations and permit certain operations on them, comprise each screen. A schematic of one of these screens is shown in Figure 3. As the programmer and user define the space, it is also possible to use different organizational methods for the space of alternatives. In Figure 4, for example, one sees 3 different ways to organize a space with three dimensions. Using the terminology of Figure 3, the representatives $x_1 \dots x_4$ in Figure 4(b) are formed by choosing sequentially from X and randomly from Y and Z.

The user indicates desirable elements or complete visual representations by non-verbal selection (done by clicking directly on the desired cell). Once the user is satisfied with the selections made on a particular space, a new space consistent with those selections is generated by a genetic approach which performs crossover operations amongst selected combinations. Successive generations can be used to either narrow or expand the search space (up to the size of the original), depending on the needs of the user. Additionally, an "image editor" is provided to directly make small changes. In this way, the space of all available visual representations can be navigated.

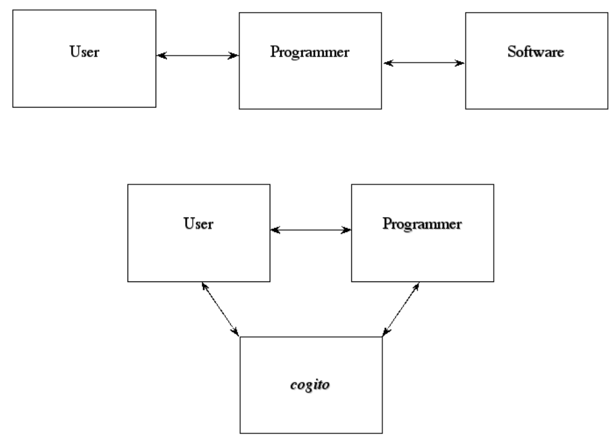

Figure 2: In the traditional model of interaction with visualization systems, the programmer mediates the user's experience with the software. The new model embodied in *cogito* allows the user to work directly with the software.

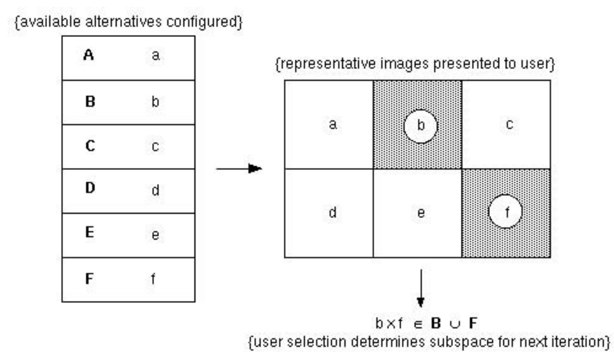

Figure 3: Schematic look at the interface: the space of available alternatives is grouped according to user-specified criteria. Each group (A – F) has a representative element (a – f) which is displayed to the user. The subspace for the next search iteration is based on the user selection (b and f).

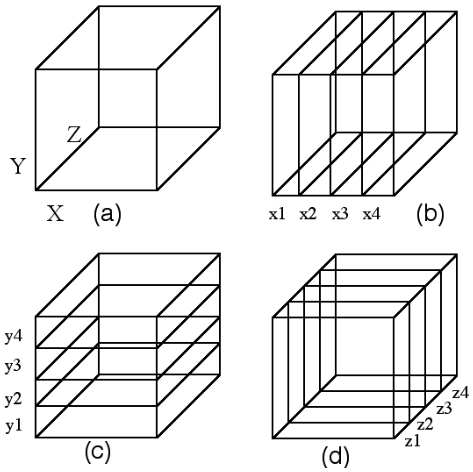

Figure 4: Consider a three-dimensional space, depicted in the top left, with axes X, Y, and Z. Organizing the space in terms of any of those 3 axes leads to the other states depicted. If elements in component X are chosen sequentially, those in Y and Z can be selected randomly to give a sense available options.

3. Computational Support for Discovery

A fundamental original emphasis for the *cogito* system was support for the user *without* directing the user. In very large and complex information spaces, this emphasis is perhaps impractical or even unwanted when considering experts in some domain of knowledge. There are two basic ways in which some direction can be given to the *cogito* user.

The first may emerge as patterns in usage of the *cogito* system over time. For the same data set and collections of available visual representations, it might be very instructive to know if previous users had selected certain visual representations in certain cases. This type of information can be garnered by multidimensional scaling [16], for example.

The second type of direction can derived from the data to be analyzed, where algorithms for learning can be applied. These learning methods are outlined in the balance of this section.

Learning from examples, known as inductive learning, is perhaps the oldest and best understood problem in artificial intelligence [43]. For our purposes, we may view this problem as indentifying the documents which a user is likely to be interested in. An inductive algorithm can, from a given sample of documents classified as relevant or nonrelevant, infer and refine decision rules.

Quinlan [24] proposed an inductive algorithm, called *ID3*, based on the statistical theory of information proposed by Shannon. Since it is essential to have an effective attribute selection criterion, ID3 uses the entropy function in selecting a suitable subset of attributes to construct a decision tree.

We illustrate this method using an example in [43]. In the following example, our problem of identifying the documents relevant to a user is synnonamous with the problem of a physician diagnosing patients. Consider the sample data in Figure 5 representing the diagnosis of eight patients by a physician. As already mentioned, the task at hand is to determine which attribute out of Height, Hair, and Eyes is the best classifier. By using the entropy function, it can be verified that attribute Hair is the best classifier. Thus, we construct the initial decision tree in Figure 6.

	Height	Hair	Eyes	Expert Classification
$o1$	Short	Dark	Blue	$-$
$o2$	Tall	Dark	Blue	$-$
$o3$	Tall	Dark	Brown	$-$
$o4$	Tall	Red	Blue	$+$
$o5$	Short	Blond	Blue	$+$
$o6$	Tall	Blond	Brown	$-$
$o7$	Tall	Blond	Blue	$+$
$o8$	Short	Blond	Brown	$-$

Figure 5: Sample data consisting of eight people diagnosed by a physician.

Any leaf node of the tree containing objects belonging to different expert classes requires further classification. In the initial decision tree, the leaf node for Blond needs refinement. Again, the objective is to determine which

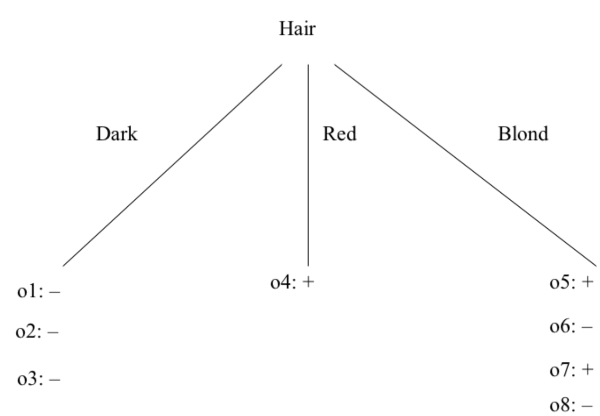

Figure 6: The initial decision tree with Hair as the root node.

of the remaining attributes best classifies the four objects $o5, o6, o7, o8$. It can be established that attribute Eyes is a better classifier than attribute Height. Thus, the initial decision tree in Figure 6 is refined as shown in Figure 7. Since each leaf node in the refined decision tree contains objects of the same expert class, no further refinement is necessary. Thus, given the sample data in Figure 5, the ID3 algorithm will produce the decision tree in Figure 7.

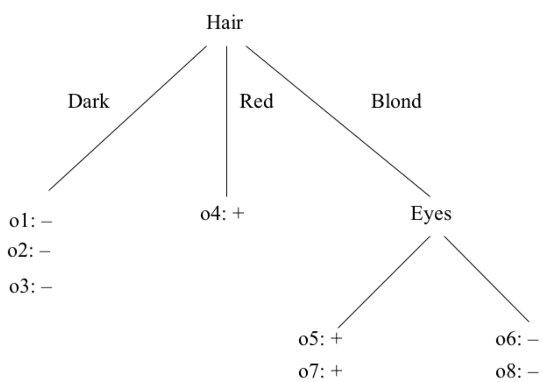

Figure 7: Refining the decision tree in Figure 6 by adding the node (classifier) Eyes.

The important point to remember is that, by providing the user with a decision tree tool, the user will be able to visualize how the attributes classify the sample documents. This information is useful as it can be used to assign a *weight* to the attributes (keywords) in future searches.

Yao [45] has recently argued that an information retrieval systems must provide a *variety* of tools to help a user understand collected data. In this section, we turn our attention to learning a probabilistic network from data. Such a tool is useful as it reflects the *probabilistic independencies* holding in the data.

Our implemented system [42] for learning a probabilistic network from data requires no à priori knowledge regarding the semantics of the attributes (variables) involved. The required input is simply a repository of observed data in tabular form. Our system is capable of learning conditional independencies from the data and outputs a probabilistic schema encoding all the discovered independencies. Due to lack of space, however, we assume the reader is familar with the notions of *probabilistic conditional independence* [41], *Markov networks* [40], and *learning algorithms* [42].

Given a distribution p_X on $X \subseteq R$, we can define a function as follows:

$$\mathcal{H}(p_X) = -\sum_{t_X} p_X(t_X) \log p_X(t_X) \quad (1)$$

$$= -\sum_{x} p_X(x) \log p_X(x), \quad (2)$$

where t is a tuple (configuration) of X and $x = t_X = t[X]$ is a X-value, and $\mathcal{H}$ is the Shannon entropy.

Let $\mathcal{G} = \{R_1, R_2, \ldots, R_n\}$ be hypertree and $R = R_1 R_2 \ldots R_n$. Let the sequence R_1, R_2, $\ldots R_n$ be a tree construction ordering for $\mathcal{G}$ such that $(R_1 R_2 \ldots R_{i-1}) \cap R_i = R_{i^*} \cap R_i$ for $1 \leq i^* \leq n-1$ and $2 \leq i \leq n$. A joint probability distribution p_R factorized on $\mathcal{G}$ is a Markov distribution, if and only if

$$\mathcal{H}(p_R) = \sum_{i=1}^{n} \mathcal{H}(p_{R_i}) - \sum_{i=2}^{n} \mathcal{H}(p_{R_{i^*} \cap R_i}). \quad (3)$$

This theorem indicates that we can characterize a *Markov distribution* by an entropy function. We now demonstrate how we learn the dependency structure of a Markov distribution.

Initially, we may assume that all the attributes are probabilistically independent, i.e., there exists no edge between any two nodes (attributes) in the undirected graph representing the Markov distribution. Then an edge is added to the graph subject to the restriction that the resultant hypergraph must be a hypertree. The undirected graph of the Markov distribution with minimum entropy is being selected as the graph for further addition of other edges. This process is repeated until a predetermined *threshold*, which defines the rate of decrease of entropy between successive distributions, is reached. From the output hypertree, we can infer the probabilistic conditional independencies which are satisfied by the distribution.

Suppose we have a database D consisting of the observed data of a set of four attributes, $\mathcal{N} = \{a_1, a_2, a_3, a_4\}$, containing five tuples as shown in Figure 8. We have set the threshold to zero, the maximum size of a clique $\eta = 4$, and the maximum number of lookahead links to one. The output is the undirected graph $\{(a_1, a_2), (a_1, a_3), (a_1, a_4)\}$. By applying the separation method, we know the following CIs hold in the observed data $\{I(a_2, a_1, a_3 a_4), I(a_3, a_1, a_2 a_4), I(a_4, a_1, a_2 a_3)\}$.

It should be mentioned that there are numerous methods for learning a *Bayesian network* [23] from data (see [42] for references). Since a Bayesian network is defined on a *directed acyclic graph*, the directionality of the edges may be interpreted as causality between variables. Thereby, one may choose to learn a Bayesian network as well as a Markov network from the sample data.

a_1	a_2	a_3	a_4
0	0	0	0
1	1	1	0
1	1	0	0
0	0	1	0
2	1	1	1

Figure 8: Observed data consisting of 4 attributes and 5 tuples.

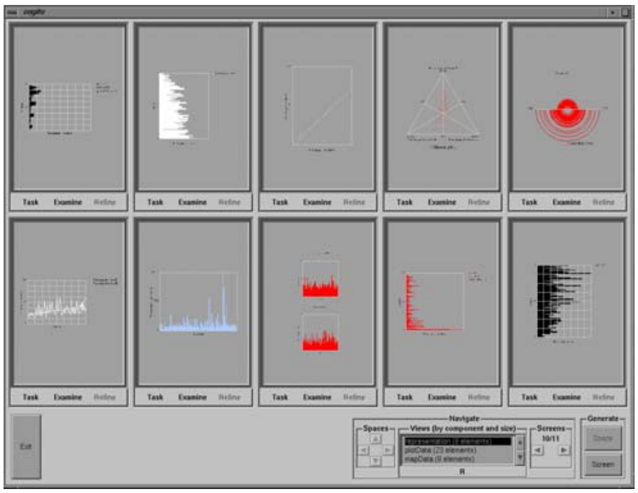

Figure 9: The interface to the *cogito* system. The interface displays a subset of available representations (sampled according to the selected organization of the search space), generated from the current data, with which the user can interact.

4. Integration

The *cogito* interface is shown in Figure 9. An integrated visualization and discovery tool will definitely help to address the issues of trust [32], as users can manipulate data and represent them in a variety of ways. The variety of methods available to the user will enable them to arrive at conclusions from several means.

Rather than lose sight of the forest for the trees, this approach allows users to examine the trees in the context of the forest, and to examine the forest at various levels of granularity, according to different criteria.

For visualization and discovery, we see two important advantages arise. Interesting patterns can be found through visualization which can be then coded within the discovery portion and similarly, patterns discovered by the learning algorithms can focus the interpretation efforts in the visualization stage. Thus, we feel our system coincides nicely with the architecture of web-based support systems suggested by Yao and Yao [44].

5. Future work

The modelling and representation of the space of available alternatives has proven to be important in many respects. Primarily, it provides each user with the means to explore in a safe, structured environment that can then act as a record of the whole decision process. The approach is even generalizable beyond the scope of the visualization and discovery. Work is now being done to use this paradigm in numerical experimentation. Consider using this approach to manage a numerical experiment where each parameter becomes a component and the values for that parameter become elements. We are pursuing other possible applications for this paradigm.

References

[1] B. L. Allen. *Information Tasks: Toward a User-Centered Approach to Information Systems*. Academic Press, 1996.

[2] J. Bertin. *Semiology of graphics : diagrams, networks, maps*. University of Wisconsin Press, 1983. translated by W. J. Berg.

[3] S. M. Casner. A task-analytic approach to the automated design of graphic presentations. *ACM Transactions on Graphics*, 10(2):111–151, April 1991.

[4] E. H. Chi. Principles for information visualization spreadsheets. *IEEE Computer Graphics and Applications*, pages 30–38, July/August 1998.

[5] D. S. Dyer. A dataflow toolkit for visualization. *IEEE Computer Graphics and Applications*, pages 60–69, July 1990.

[6] G. Fischer and B. Reeves. Beyond intelligent interfaces: Exploring, analyzing, and creating success models of cooperative problem solving. *Journal of Applied Intelligence*, 1:311–332, 1992.

[7] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, Reading, MA, 1989.

[8] N. Grady et al. Integrating data mining and visualization processes. In U. Fayyad, G. Grinstein, and A. Wierse, editors, *Information Visualization in Data Mining and Knowledge Discovery*, pages 299–303. Morgan Kaufmann, 2001.

[9] R. Hamming. *Numerical Methods for Scientists and Engineers*. McGraw-Hill, 1962.

[10] D. H. Hepting. *A New Paradigm for Exploration in Computer-Aided Visualization*. PhD thesis, Simon Fraser University, 1999. Ph.D. Dissertation.

[11] D. H. Hepting. Towards a visual interface for information visualization. In E. Banissi, editor, *Proceedings of the Sixth International Conference on Information Visualization*, pages 295–302. IEEE Computer Society, 2002.

[12] M. E. Jessup. Scientific visualization: Viewpoint on collaborations of art, science, and engineering. *SIGBIO Newsletter*, pages 1–9, February 1992.

[13] S. Kochhar, M. Friedell, and M. LaPolla. Cooperative, computer-aided design of scientific visualizations. In *Proceedings of Visualization '91*, pages 306–313, 1991.

[14] S. Kochhar, J. Marks, and M. Friedell. Interaction paradigms for human-computer cooperation in graphical-object modelling. In S. MacKay and E. M. Kidd, editors, *Proceedings of Graphics Interface '91*, pages 180–189, 1991.

[15] J. R. Koza. *Genetic Programming : On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*. MIT Press, 1992.

[16] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1–29, 1964.

[17] M. Levoy. Spreadsheets for images. In A. Glassner, editor, *Computer Graphics: SIGGRAPH 94 Proceedings*, pages 139–146, 1994.

[18] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5(2):110–141, 1986.

[19] J. Marks et al. Design Galleries: A general approach to setting parameters for computer graphics and animation. In *SIGGRAPH '97 Conference Proceedings*, pages 389–400, 1997.

[20] B. H. McCormick, T. A. DeFanti, and M. D. Brown. Visualization in scientific computing. *Computer Graphics*, 21(6), November 1987.

[21] S. D. Neill. The reference process and the philosophy of karl popper. *RQ*, pages 309–319, Spring 1985.

[22] D. A. Norman. *The Psychology of Everyday Things*. Basic Books, New York, 1988.

[23] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Francisco, California, 1988.

[24] J. Quinlan. *Machine Learning: the Artificial Intelligence Approach*, chapter Learning Efficient Classification Procedures and Their Application to Chess End Games. Tioga Press, Palo Alto, 1983.

[25] H. W. J. Rittel. Second-generation design methods. In *Developments in Design Methodology*, pages 317–327. Wiley and Sons, 1984.

[26] B. E. Rogowitz and L. A. Treinish. Data structures and perceptual structures. *SPIE*, 1913:600–612, 1993.

[27] S. F. Roth, J. Kolojejchick, J. Mattis, and M. Chuah. Sagetools: An intelligent environment for sketching, browsing, and customizing data-graphics. In *Proceedings CHI'95 Human Factors in Computing Systems*, pages 409–410. ACM Press, 1995.

[28] W. J. Schroeder, K. M. Martin, and W. E. Lorensen. The design and implementation of an object-oriented toolkit for 3D graphics and visualization. In *Proceedings of Visualization 96*, pages 93–100, 1996.

[29] H. Senay and E. Ignatius. A knowledge-based system for visualization design. *IEEE Computer Graphics and Applications*, 14(6):36, 1994.

[30] K. Sims. Artificial evolution in computer graphics. In R. J. Beach, editor, *Computer Graphics: SIGGRAPH '91 Conference Proceedings*, pages 319–328. ACM Press, 1991.

[31] P. Skagestad. Thinking with machines: Intelligence augmentation, evolutionary epistemology and semiotics. *Journal of Social and Evolutionary Systems*, 16(2):157–180, 1993.

[32] K. Thearling et al. Visualizing data mining models. In U. Fayyad, G. Grinstein, and A. Wierse, editors, *Information Visualization in Data Mining and Knowledge Discovery*, pages 205–222. Morgan Kaufmann, 2001.

[33] S. Todd and W. Latham. *Evolutionary art and computers*. Academic Press, London, 1992.

[34] J. W. Tukey. *Exploratory data analysis*. Addison-Wesley, 1977.

[35] C. Upson et al. The application visualization system: A computational environment for scientific visualization. *IEEE Computer Graphics and Applications*, 9(4):30–42, 1989.

[36] P. F. Velleman and P. Pratt. A graphical interface for data analysis. *Journal of Statistical Computation and Simulation*, 32:223–228, 1989.

[37] J. von Neumann. Recent theories of turbulence. In A. Taub, editor, *Collected Works of John von Neumann*, volume 6, pages 437–472. MacMillan, New York, 1963.

[38] P. Wegner. Why interaction is more powerful than algorithms. *Communications of the ACM*, 40(5):80–91, 1997.

[39] T. Winograd and C. F. Flores. *Understanding Computers and Cognition*. Ablex, Norwood, New Jersey, USA, 1985.

[40] S. Wong and C. Butz. Constructing the dependency structure of a multi-agent probabilistic network. *IEEE Transactions on Knowledge and Data Engineering*, 30, Part A(6):785–805, 1999.

[41] S. Wong, C. Butz, and D. Wu. On the implication problem for probabilistic conditional independency. *IEEE Transactions on Systems, Man, and Cybernetics*, 30, Part A(6):785–805, 2000.

[42] S. Wong, C. Butz, and Y. Xiang. Automated database scheme design using mined data dependencies. *Journal of the American Society for Information Science*, 49(5):455–470, 1998.

[43] S. Wong, W. Ziarko, and R. Ye. Comparison of rough-set and statistical methods in inductive learning. *International Journal of Man-Machine Studies*, 24:53–72, 1986.

[44] J. T. Yao and Y. Yao. Web-based support systems. In *Proceedings of the WI/IAT 2003 Workshop on Applications, Products, and Services of Web-based Support Systems*, 2003.

[45] Y. Yao. Information retrieval support systems. In *Proceedings of the 2002 IEEE World Congress on Computational Intelligence*, pages 1092–1097, 2002.