01 Jan 1991

A Visualization Model For Massively Parallel Algorithms

Rashi Khanna

Bruce M. McMillin
Missouri University of Science and Technology, ff@mst.edu

A VISUALIZATION MODEL FOR MASSIVELY PARALLEL ALGORITHMS

Rashi Khanna and Bruce McMillin‡
Department of Computer Science - University of Missouri-Rolla
Rolla, MO 65401 - *ff@cs.umr.edu*

Abstract

A visualization model has been developed to analyse the performance of a massively parallel algorithm. Most visualization tools that have been developed so far for performance analysis are based generally on individual processor information and communication patterns. These tools, however, are inadequate for massively parallel computations. It is difficult to comprehend the visual information for many processors. The model, SMILI (Scientific visualization in Multicomputing for Interpretation of Large amounts of Information), addresses this problem by using abstract representations to attain a composite picture which gives better insight to the behavior of the algorithm. Chernoff's Faces have been selected to represent the multidimensional data because of their ability to portray multidimensional data in a very perceptible manner. SMILI has been used on an asynchronous massively parallel PDE (partial differential equation) solver that is based on the multigrid paradigm. The visualization tool helps in tuning the control parameters of the multigrid algorithm to get optimal results.

1. Introduction

Visualization of scientific data sets plays an important role in understanding complex phenomena. The motivation for visualization is in the fact that it is tool for discovery and understanding. Experiments and numerical simulations of physical processes tend to yield high-resolution multivariate data. These incredibly rich and complex data sets consist of several quantities. When the data are dependent on time or one or more parameters, the dimensionality and the size of these data sets increases rapidly. Extensive interaction between experimental and numerical analysis is necessary both to verify the theoretical models used in the simulations and to tune parameters. Visualization of the data helps here by assisting the user in extracting qualitative information and in making quantitative comparisons to iteratively optimize the system. However, most of the large multivariate data sets are not suited to direct display (imagine a million points color-coded in space!). Analysis is almost always required. Unless the display is founded on analysis or interpretation relevant to the problem, you end up with a pretty picture with little scientific value.

Recently there has been growing interest in performance visualization of parallel programs being executed on multiprocessor systems. Understanding why a parallel algorithm performs poorly is a difficult task. Complex dynamic interactions take place that can seriously degrade expected performance. An appreciation for what is happening during the execution in a parallel algorithm is necessary to address the inefficiencies and overheads introduced by various computation and communication structures. The old adage that *a picture is worth a thousand words* has motivated the implementation of several visualization tools that confront the problem of observing the operation of a parallel program. Visual interfaces have become nearly universal, as a result of the availability of desktop graphics, the need to represent much complex data intelligibly, and the desire to provide an intuitive interface to the user.

Performance tools can be characterized as either being hardware-related or application-oriented. Hardware-related tools generally focus on viewing parameters related to communication events and processor activity occurring in the system. While helpful, it remains to be seen how useful such information is in finding and rectifying the causes of poor performance in a general class of problems. Application-oriented tools, on the other hand, provide displays for parameters directly dependent on the application. Some of the currently existing tools [3] are: Parasoft's PM, Hyperview, Axe, Poker, Paragraph, and Triplex. SMILI, the visualization tool that is the subject of this paper, displays information in the context of the application. SMILI is hardware independent and can be used to analyze programs designed for different architectures. Most importantly, it has a novel way of scalably displaying complex data. Most of the tools that have been designed [3] cater to a relatively smaller level of parallelism and the displays have been implemented for 8 to 16 node machines. The question of scalability to much bigger machines is then raised. Certainly, not much can be extracted if one has to scan 128 point kiviat graphs (Figure 1) or a gantt chart displaying processor activity in 128 processors. Human perception is limited and the whole purpose of visualization is forfeited if quick and clear behavioral patterns are

‡ This work was supported in part by the National Science Foundation under Grant Numbers MIP-8909749 and CDA-8820714, in part by the AMOCO Faculty Development Program, in part by the Manufacturing Research and Training Center (MRTC), and in part by the McDonnell Douglas Corporation.

0-8186-2290-3/91/0000/0617/$01.00 © 1991 IEEE

not perceived by just looking at and comparing the pictures. SMILI addresses the problem of scalability by using abstract graphical techniques for representation of multivariate data. It is capable of dealing with performance data generated from massively parallel algorithms by treating the data as a set of multivariate points in space. The performance measures for parallel algorithms are inter-related and with SMILI it is possible to detect the dependencies. Conclusions of the performance maybe drawn only after looking at a global picture. Hence, multivariate abstractions are sought.

2. Method and Limitations

Since we are dealing with multivariate data that is subject to strong and complex relationships, the objective is to represent the data in a way that an observer can quickly comprehend relevant information. Most people are aware of the methods for visual representation of univariate data, for example histograms, frequency polygons etc. Methods for visual representation of multivariate data are less common and less well known. Review of multivariate display techniques [6, 7, 8], such as those of Figure 1, indicates that they generally involve the use of a symbol to represent high-dimensional data.

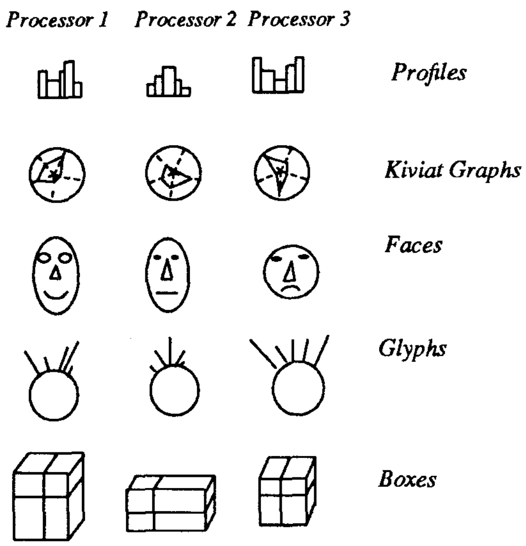

Figure 1. Multivariate Representations for Processor Information

Chernoff [6], in particular, describes a technique which represents each p-dimensional observation by the cartoon of a face whose features, such as the shape of the nose and the curvature of the mouth, are governed by the values taken by particular variables. A sample of points in p-dimensional space is represented by a collection of faces. The ability to relate faces to emotional reactions seems to carry a mnemonic advantage. Certain major characteristics of the faces are instantly observed and easily remembered in terms of emotions and appearance. Finer details and correlations become apparent after studying the faces for a time. The awareness of these does not drive out of mind the original major impressions. The major advantage to be derived from using the faces should be in the heightened qualitative awareness of which numerical calculations are relevant.

Faces are effective in revealing rather complex relations in the multivariate data. The study of faces does not seem to become more difficult as the number of variables increases. In fact, the information content transmitted seems to be richer as the number of variables containing useful information increases. The Chernoff face method has several distinct advantages over other representational techniques. First, faces are easily recognized and described. We grow up studying faces and learning to recognize different facial expressions. Secondly, we are able to link facial characteristics with the physical meaning of the variables. The smile can be used to represent a "success/failure" variable, the eyes can represent a "slyness" variable, etc. In spite of the above plusses for using Chernoff's faces, there are some minuses. The faces can be abused, they can be used to portray wrong information. The built-in dependencies among facial features may distort the data representation enough to cause erroneous impressions. Thus, tuning of the faces is important.

SMILI adopts the faces technique for the performance of data representation. To tune the algorithm, the behavior of certain computable quantities need to be observed. These performance quantities constitute the variables of the multivariate representation and are assigned to the different features of the Chernoff's face. The faces appear on the screen dynamically and change their features as the animation of the algorithm proceeds. In this way, one can see exactly at what stage of the algorithm the malfunctioning occurs. Also if individual processor information is desired as in most other tools, a face can be used to display each processor's statistics. So, instead of scanning multiple displays to make comparisons of different processor activities, single face representations can be studied. Faces are definitely simpler and more effective in relating the same amount of information.

3. Model Problem

SMILI was used to tune the asynchronous multigrid algorithm to determine the optimal control parameters. The details of the asynchronous multigrid algorithm, PAFMV, are provided in [10]. Within the algorithm, information exchange occurs non-deterministically. It is possible that at intermediate stages of the program, the solution which is being computed diverges instead of converging, because of the non-deterministic behavior. These intermediary hindrances can deteriorate the performance of the algorithm. Tuning the algorithm using SMILI assists in determining those control parameters that will minimize the hindrances and will lead to a fast convergent solution.

4. Implementation

4.1 Data Collection

The PAFMV [10] multigrid algorithm on a 16 processor Intel iPSC/2 is used for data generation. Before the performance visualization tool can be applied, the relevant performance data needs to be collected. For this purpose, trace files are generated from each processor as the program executes. The resulting event traces can then be used to compile summary statistics that provide a global view of the performance.

4.2 Visual Interface

SMILI is based on the X Window System, and thus runs on a variety of workstations. SMILI currently provides five displays, each of which may be selected from the control menu. The first display replays the execution of the algorithm. Three grids appear in one part of the screen representing the three grids of the multigrid algorithm. A spectrum of shades from black to white appear on each grid. Each color reflects a certain value of the solution on that grid, with white representing the highest value and black the lowest. The second display is of the Chernoff's face which changes expressions as the algorithm proceeds, while depicting performance information. The third option couples the two displays allowing the user to view the behavior of the solution as well as the performance parameters. The fourth display is a static display of a face depicting post-mortem results, meaning it displays performance parameters whose values can be computed only after the program completes execution (eg. processor utilization). The fifth display is a history trace showing all the faces that appeared in the performance window.

5. Performance Monitoring

5.1 The Tuning Process

Various experiments were conducted to determine the optimal control parameters of the PAFMV algorithm which are illustrated in Figure 2. To create a Chernoff's face, an assignment of the performance parameters to the facial features is made. The assignment maybe random or deliberate. Some users prefer the random assignment to reduce the subjective elements, others deliberately employ perception of facial characteristics in the assignment. If a change in any of the control parameters affects the performance, it will be reflected in the facial expression of the Chernoff's face. The mapping is as follows: the *outline* of the face is mapped to the residual norm [10] that is computed from the trace data, the *curvature* of the mouth is mapped to the error norm [10], the rate of change in the residual norm is reflected in the *shape of the eyes*, the rate of change of the error norm is reflected in the *slant of the eyebrows*, and the execution time is mapped to the *nose height*.

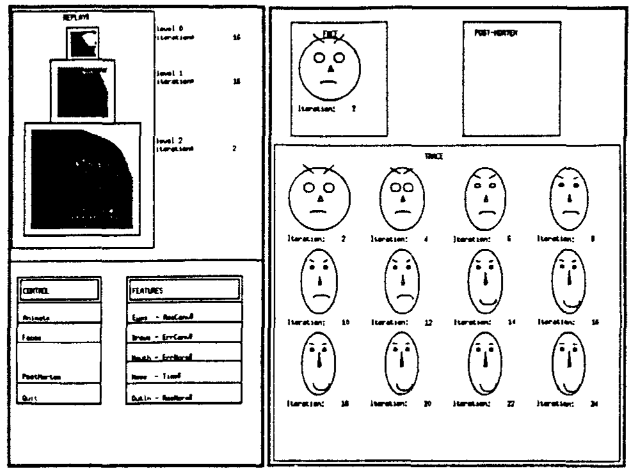

Figure 2. SMILI's visual interface

Given the mappings of the features, one has to know what facial expressions to look for in midst of so many different expressions. With the above assignment, a narrow face with a big smile is sought as that represents a low value for the two norms. While comparing two sets of faces, the slant of the eyebrows is observed for determining the amount of change in the error norm as the iterations proceed. The longer the slant remains before convergence is achieved, the greater is the rate of convergence for the error. The similar is true for the eyes. The longer the eyes remain wide before convergence is achieved, the higher is rate of convergence of error norm. It should be noted at this point, that while studying the faces, individual features are not always scanned, but the overall expression is studied. After using the faces for sometime, one "learns" the expressions and can relate easily to its interpretation.

To tune the parameters, firstly a random choice is made. Only one of the parameters is then varied, keeping the other four fixed, and the performance is observed. After determining approximately, the optimal value for that fixed parameter, another random set of five parameters is chosen and the process is repeated. The optimal value obtained from the second set of experiments is then compared with the first. If they match, then the optimum has been found. If not, more experiments are conducted and results are compared. The whole process is repeated for the rest of the four parameters. Once an optimal value has been determined for one of the parameters, it is used in the consecutive experiments.

The following figure (Figure 3) illustrates one of the traces studied during tuning. As the number of iterations proceeds, the faces become narrower and start to

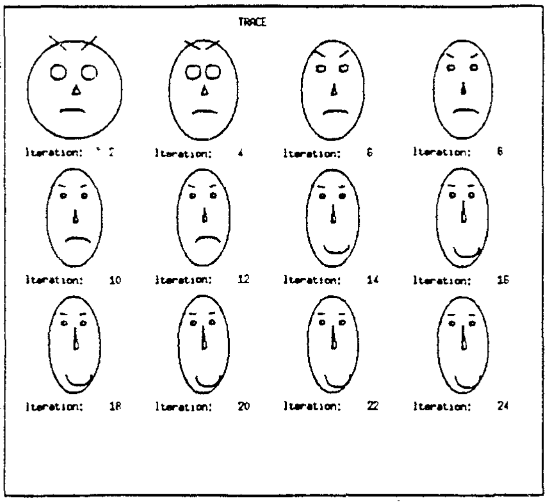

Figure 3. A History Trace of Faces

smile. While making comparisons with other faces, one looks for this type of convergent behavior. Also important in the comparisons, is the time the faces take in a particular experiment to arrive at a convergent solution; how many iterations do the faces take before they attain a curvy smile and narrow width?

5.2 Analysis

The previous section demonstrates the tuning process of the control parameters. It is important to note that although the information regarding performance lies in the characteristics of the individual features, the face is viewed as a whole while assessing the performance. The overall expression is the observer's own synthesis of the various individual features. It constitutes a single image depicting the overall position of the point in its multidimensional space. The ability to instantly relate to the facial expression is acquired through the process of learning the faces.

In the first set of experiments, after the assignment of variables to features is done, the faces are analyzed by observing the variation in the features. After some experiments, the observer gets accustomed to seeing the faces and begins to distinguish the "good" faces from the "bad" faces. In the subsequent experiments, performance comparisons amounts to simple pattern recognition. This is helpful while tuning because a change in a certain parameter results in a change in the facial expressions. If the change in the facial expression is positive, then it is known that the change in the parameter was a valid one. In the above set of experiments, a narrow smiling face is considered desirable. The decision of whether a certain set of parameters performs better compared to another set of parameters is based on whether a narrow smiling face is attained or not and how soon it takes to reach that state. A bad set of parameters is then easily distinguishable if the facial expressions start diverging, implying that instead of approaching the narrow smiling face, either the face starts to frown or it widens inconsistently.

The faces seem vulnerable to the way in which features are assigned to the variables. It is feared that because of the "emotional" reaction it generates, a different assignment may affect the analysis. For example, if a frown on a face actually implies good behavior, the face maybe overlooked because humans relate to frowns negatively. Although this subjectivity is often criticized, it can be considered as a positive feature rather than a drawback because a proper assignment may actually enhance the analysis procedure.

6. Conclusion

A framework has been developed to provide scalable performance visualization for parallel programs, and to assist in optimizing application performance. SMILI uses cartoon faces to represent the multivariate performance data. Faces have been chosen over other representations because of their ability to portray information in a very perceptible manner.

7. References

[1] G. Nielson, and B. Shriver, *Visualization in Scientific Computing*, IEEE Computer Society Tutorial, 1990.

[2] T. Lehr, Z. Segall, E. Caplan, et al., *Visualizing Performance Debugging*, Computer, Vol. 22, No. 10, Oct. 1989, pp. 38-51.

[3] K. Nichols, *Performance Tools*, IEEE Software, Vol. 7, No. 3, May 1990, pp. 21-30.

[4] M. Heath, *Visual Animation of Parallel Algorithms for Matrix Computations*, Proceedings of the Fifth Distributed Memory Computing Conference, Vol. 2, 1990, pp. 1213-1222.

[5] D. Rover, and C. Wright, *Pictures of Performance: Highlighting Program Activity in Time and Space*, Proceedings of the Fifth Distributed Memory Computing Conference, Vol. 2, 1990, pp. 1228-1233.

[6] H. Chernoff, *The Use of Faces to Represent Points in a k-Dimensional Space Graphically*, Journal of the American Statistical Association, Vol. 68, No. 342, Jun. 1973, pp. 361-368.

[7] B. Everitt, and P.Nichols, *Visual Techniques for Representing Multivariate Data*, The Statistician, Vol. 24, No. 1, pp. 37-49.

[8] P. Wang, *Graphical Representation of Multivariate Data*, Academic Press, 1978.

[9] D. Ferrari, G. Serazzi, and A. Ziegner, *Measurement and Tuning of Computer Sytems*, Prentice Hall, 1983.

[10] S. Franks, B. McMillin, and R. Khanna, *PAFMV - Paiwise Asynchronous Multigrid*, Proceedings of the 1990 Intl. Conference on Parallel Processing, pp. I388-I392.