Grid Monitoring Data Analysis with 3D graphics

Lucas Mello Schnorr, Philippe O. A. Navaux
Instituto de Informática
Universidade Federal do Rio Grande do Sul
CEP 91501-970 - Porto Alegre - RS - Brazil
E-mail: {lmschnorr,navaux}@inf.ufrgs.br

Guillaume Huard
Laboratoire LIG - MOAIS INRIA Project
ENSIMAG - 51 avenue Jean Kuntzmann
38330 - Montbonnot Saint Martin - France
E-mail: Guillaume.Huard@imag.fr

Abstract

This paper presents Triva, a three dimensional visualization prototype conceived to the analysis of monitoring data collected in distributed environments. The basic idea of this prototype is to apply information visualization techniques in the context of parallel processing in order to render 3D graphics that enable a better understanding of parallel applications and resources. This paper presents three aspects of Triva: the idea of the 3D visualization applied to distributed environments; the details of the prototype developed, focusing in the way it can be extended, and finally the results obtained with it, showing an analysis of a parallel application executed in a Grid with the KAAPI communication library.

1. Introduction

The analysis of parallel applications is an important task in a distributed environment, such as Grids. This task allows a developer to improve the performance of the application by possibly taking into account environment characteristics and correlating them to the application behavior. This operation is usually a complicated task, considering the amount of data to analyze and the lack of determinism in the execution of parallel applications.

The use of visualization is one solution to perform this analysis. Most of these tools work with two dimensional graphics, where one of the dimensions is used to describe the monitored entities of the application, such as processes, threads, and so on, and the other axis is used to represent time. With this, developers can visually analyze the time evolution of application components through graphics commonly known as Gantt-charts [15]. Examples of this kind of tool are Pajé [4], Paradyn [8], Paraver [9] and many others. Although gantt-charts is widely used in distributed environments to analyze parallel applications, it has a set of limitations when applied to more dynamic and scalable environments. Such limitations include the limited set of interaction mechanisms available to the user, the use of only one dimension to describe the monitored entities and the lack of support to analyze the application correlated with resources monitoring data.

These limitations in traditional visualization schemes for monitoring data lead us to propose a novel way to analyze monitoring data from parallel applications. Our approach is to use a three dimensional (3D) visualization. With this, we preserve two dimensions to describe the application's entities and use the third dimension to observe the evolution of the data over time [11]. Additionally, we intend to use information visualization techniques in this 3D scene in order to improve the perception of application's characteristics and patterns.

Some tools already employ a 3D visualization to analyze monitoring data. OpenMosix [2] has one tool called 3dmosmon that shows the use of resources in real time. Virtue [13] offers three types of 3D graphics for the application analysis: widearea, time-tunnel and call-graph. Wide area uses a geographic map, like the one available in GridPP [6]. Time-tunnel shows the application's time evolution inside a 3D cylinder and call-graph shows 3D objects to illustrate the function call order of the application. The comparison with our approach shows that the only similar work is the time-tunnel display in Virtue. However, this display does not offer multiple configurations for the visualization base, as we do with our model.

This paper presents an overview of our 3D visualization and its corresponding implementation through our prototype Triva, in the next section. We also present Triva's screenshots in section 3. After this, we conclude the paper with a summary and future work.

2. Triva Approach

This section presents the prototype Triva and the 3D approach [11]. The main characteristics of the idea are the use of three dimensions to represent monitoring data collected in distributed environments, its ability to aggregate information visualization techniques to the analysis of parallel applications and an aesthetic view [7] of the data. We give an overview of the 3D model focusing in its extensible feature and relate it to the concepts of parallel processing. The second subsection explain the implementation details of Triva. It presents the decisions we took to implement the model and the interaction with existing software libraries.

2.1. 3D Model

Traditional visualization tools for monitoring data [4, 8–10] usually offer two-dimensional graphics. In most of these tools, one of the dimensions is used to describe the resources and their organization and the other one is used to show the evolution of these resources over time. This kind of graphic is commonly named as Gantt-charts [15].

Another type of two dimensional visualization is the one used by online monitoring systems, such as Monika of Grid5000 [3], where both dimensions are used to describe the resources. In these systems, the visualization shows only the current state of the resources, and does not provide a way to visualize historical information.

The proposed three dimensional approach tries to unify the idea of having two dimensions to describe the resources, together with a third dimension to show historical information. With this, different techniques, such as graph drawing and squarified treemap, can be applied to better visually represent the resources and their correspondence to the applications being executed. Figure 1 shows a synthetic visualization of the 3D model. The monitored entities are placed in the visualization base, and their state changes over time are drawn in the vertical axis. In the hypothetical example of the figure, a network graph was drawn in the visualization base together with an application with five processes running on top of that network. By using information visualization and interaction techniques within the three dimensional scene, the user can better correlate the application with the used resources.

The monitored data that can be visualized in this approach is basically composed by events that describe the behavior of monitored entities. These entities can be anything from an application process or thread, to a single resource, such as a processor, machine or the network itself. Each event has one type and is always related to a monitored entity, having additional information that corresponds only to its type. The model receives these events always as a flow. The events can also be read from trace files, where each event is read at a time and passed to the model. Figure 2 shows the component model, used to generate a 3D scene with the monitored data.

The component A is responsible to receive the monitoring data from possibly multiple sources, in a distributed system. It takes this information and send it to the component B, which is responsible to extract from the flow of events the information necessary to build a specific visualization base configured by the user. The information extracted usually is composed by a set of monitored entities, but can also be any type or set of types of events. This data is sent to one of the D components. It can have any number of components D as the number of visual techniques used to draw the data in the visualization base. Today, we planed at least four types of components D: one to dynamically generate the application graph [11]; another to draw the application data over a static network graph; and the last two to use dynamic treemaps [14], one showing the application structure and another to show the application over a resource treemap. All D components must follow a strict protocol which enables them to communicate with component C. The component C is responsible to render the 3D scene using the information from one of the components D and also the flow of events sent by component B.

The component C, seen in Figure 2, is also responsible to handle visual interaction techniques. These techniques can be used in order to perform a better analysis of the monitored data. The techniques include zooming facilities, camera translation and rotation to obtain a different point-of-view of the data, animations of the visual objects and so on.

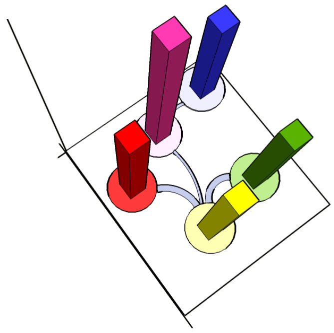

Figure 1. The three dimensional approach with a network graph represented in the base.

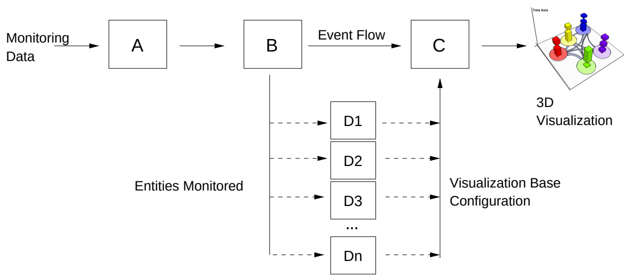

Figure 2. Component model used to create a three dimensional visualization.

The model can be easily extended with new visualization techniques. For example, in the case where the monitored data have enough information to group the application being analyzed by the resources used, a different visualization technique could be applied in order to group the entities by grids - clusters - nodes - processors and cores. Using aggregation and filtering techniques, the user can then analyze the application in different levels of detail. The next section explains how this model was implemented.

2.2. Prototype Implementation

A 3D visualization prototype called Triva is being developed in order to assess the potential of the three dimensional model for visualization of monitoring data. Triva stands for ThRee dimensional and Interactive Visualization Analysis. Its implementation uses existing tools and libraries such as DIMVisual [12], Pajé [4] and Graphviz [5] in order to provide a 3D visualization tool for monitoring data. Figure 3 depicts the implementation design in modules.

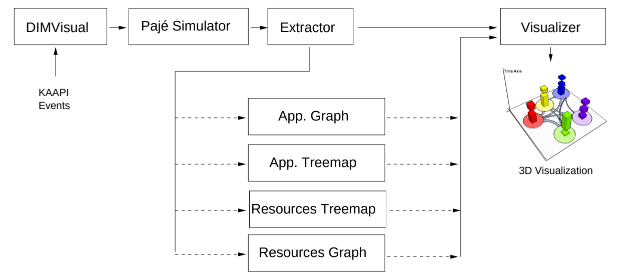

Figure 3. TRIVA implementation in modules.

Triva uses as input the monitoring data provided by the library DIMVisual. This library is able to integrate data from different sources in a distributed system. Currently, DIMVisual is configured to use only KAAPI trace files as input. In Figure 3, the DIMVisual module generates as output a flow of events in the Pajé file format. These events are received by the Pajé Simulator, which is the same component used by the Pajé Visualization Tool. The simulator generates as output a set of visual objects ready to be visualized. In Triva, we built an extractor module that selects from the flow of visual objects generated by the simulator only objects that the modules that appears under it need.

Still in Figure 3, we have four modules that are responsible for generating different visualization base configurations. The Application Graph module receives from the extractor only monitored entities, such as processes and threads, and the interactions among them. It uses the library Graphviz to generate a graph in the visualization base. The graph helps the user of Triva to detect the communication pattern of the application. Another module is the Application Treemap. It is responsible to generate a squarified treemap [1] in the visualization base only with information that the extractor sends it. This module is configurable from the user interface, allowing the user to configure the treemap with specific details from the monitoring data, such as the number of nodes, number of relations among monitored entities, the size of communications, among others. The third module is the Resource Graph. This module is configured with a file in the dot format of Graphviz and generates a graph that can represent the network topology of a distributed environment, or the connections of the middleware used by the application. The module also receives from the extractor the monitored entities, placing them in the visualization in the resource that the monitored entity used. The objective of this module in the context of application analysis is to offer the user the possibility to analyze the application in a multi-level approach, together with the graph of resources. The fourth module is the Resources Treemap module. It is configured with a file that has a hierarchical description of the resources. A usual configuration is a hierarchy of Grid - Cluster - Machine - Processors. The module cooperates with the visualization module in order to dynamically change the treemap in the visualization following the configurations done by the user. As a result, the analysis can be used in order to identify possible unbalances in the execution.

One advantage of Triva is the use of modules to generate the visualization base. This allows an easy extension of prototype features in order to add new techniques to generate the visualization base with the description of resources and monitored entities. This is possible with the use of a generic protocol by the extractor module. The use of Pajé also helps the development of a generic prototype, since its format is not attached to any kind of trace library [4]. The objects generated by the Pajé simulator are generic and can describe almost any kind of monitoring data, from applications traces to database management. This gives Triva a possibly wider range of users.

3. Experiments and Results

A KAAPI parallel application was used to generate the traces visualized by Triva. The application was designed to be a set of tasks and relations among them. KAAPI is then responsible to distribute these tasks and perform load balancing through work stealing algorithms.

Figure 4 shows the work stealing requests performed by the library during the execution of an application with 30 processes over two clusters, one of the clusters with 14 machines and another with 16. The squarified treemap visualization base shows both clusters: the right side of the corner represents the resources of one cluster, and the left the other one. Each process is allocated to one machine by the library.

The objective with the image is to see if the applications is well balanced over the resources it uses. In the figure, only the beginning of the application execution is shown. We can clearly see that the KAAPI library starts by a greater number of work stealing request in one of the clusters. Our tests in further analysis during this visualization have shown that the number of work stealing events, even those between the clusters, decreased during the application execution, but appear more frequently in the end of the application.

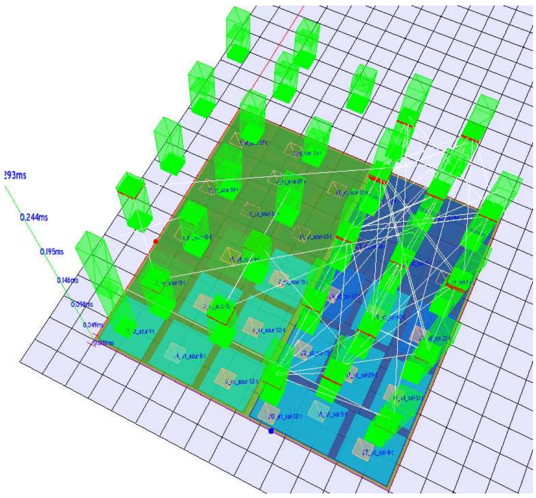

Figure 4. Resource Treemap Base

4. Conclusion

This paper presented a new approach to the visualization of monitored data from Grids. The approach consists in the use of a three dimensional visualization in order to better correlate application monitoring data with resource information [11]. It also uses information visualization techniques to improve the user experience during the application analysis. We also presented Triva, our prototype and detailed its capability of adding new visualization modules with new techniques to the visualization base. The results show that the 3D visualization can improve the user perception of application patterns specially in the cases where resource data is also presented during the analysis.

As future works, we intend to extend Triva's capabilities by offering new interaction techniques, such as animation of visual objects, time-related operations and interactive data reduction. We will also investigate the use of an animated and visual replay technique, in order to show step by step the application behavior.

References

[1] M. Bruls, K. Huizing, and J. van Wijk. Squarified treemaps. In *Proceedings of Joint Eurographics and IEEE TCVG Symposium on Visualization*, pages 33–42. IEEE Press, 2000.

[2] J. Cache. The 3dmosmon, 2007.

[3] F. Capello, B. Plateau, T. Priol, D. Vandromme, F. Cappello, M. Dayde, F. Desprez, E. Jeannot, Y. Jegou, S. Lanteri, N. Melab, R. Namyst, M. Quinson, O. Richard, and P. V.-B. Primet. Grid5000, February 2007. http://www.grid5000.fr. Last accessed February, 26, 2007.

[4] J. de Kergommeaux, B. Stein, and P. Bernard. Paje, an interactive visualization tool for tuning multi-threaded parallel applications. *Parallel Computing*, 26(10):1253–1274, 2000.

[5] J. Ellson, E. Gansner, L. Koutsofios, N. S. C., and G. Woodhull. *Graphviz Open Source Graph Drawing Tools*, volume 2265/2002, pages 594–597. Springer Berlin / Heidelberg, 2002.

[6] GridPP. Gridpp real time monitor, 2007.

[7] A. Lau and A. V. Moere. Towards a model of information aesthetics in information visualization. In *IV '07: Proceedings of the 11th International Conference Information Visualization*, pages 87–92, Washington, DC, USA, 2007. IEEE Computer Society. 0-7695-2900-3.

[8] B. P. Miller, M. D. Callaghan, J. M. Cargille, J. K. Hollingsworth, R. B. Irvin, K. L. Karavanic, K. Kunchithapadam, and T. Newhall. The paradyn parallel performance measurement tool. *IEEE Computer*, 28(11):37–46, 1995.

[9] V. Pillet, J. Labarta, T. Cortes, and S. Girona. PARAVER: A tool to visualise and analyze parallel code. In *Proceedings of Transputer and occam Developments, WOTUG-18.*, volume 44 of *Transputer and Occam Engineering*, pages 17–31, Amsterdam, 1995. [S.l.]: IOS Press.

[10] D. A. Reed, M. J. Gardner, and E. Smirni. Performance visualization: 2-d, 3-d, and beyond. In *Proceedings of IEEE International Computer Performance and Dependability Symposium IPDS-96*, page 188, 1996.

[11] L. M. Schnorr, G. Huard, and P. O. A. Navaux. 3d approach to the visualization of parallel applications and grid monitoring information. In *Proceedings of the 9th IEEE/ACM International Conference on Grid Computing*, Washington, DC, USA, 2008. IEEE Computer Society.

[12] L. M. Schnorr, P. O. A. Navaux, and B. de Oliveira Stein. Dimvisual: Data integration model for visualization of parallel programs behavior. In *CCGRID '06: Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, pages 473–480, Washington, DC, USA, 2006. IEEE Computer Society.

[13] E. Shaffer, D. A. Reed, S. Whitmore, and B. Schaeffer. Virtue: Performance visualization of parallel and distributed applications. *Computer*, 32(12):44–51, 1999.

[14] B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph.*, 11(1):92–99, 1992.

[15] J. M. Wilson. Gantt charts: A centenary appreciation. *European Journal of Operational Research*, 149(2):430–437, September 2003.