CluVis: A Framework for Cluster Computer Metadata Visualization

Ben Craig, Joseph Langley, Chris Waters*, T.J. Jankun-Kelly†

Department of Computer Science and Engineering, Mississippi State University

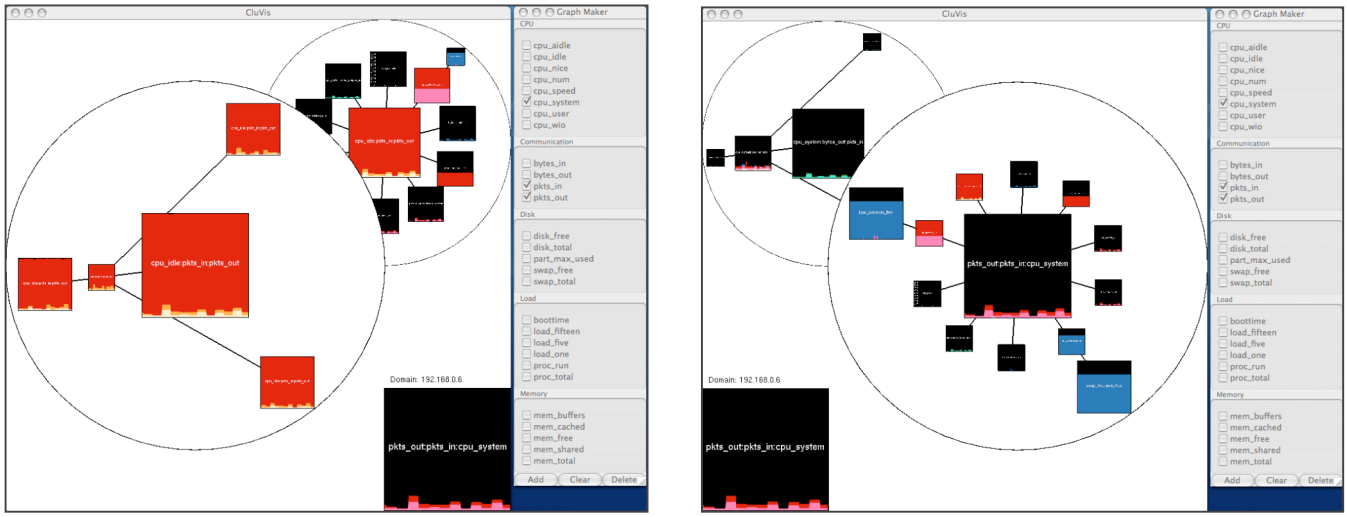

Figure 1. Left, Topology view in focus. Right, Data view in focus right after the chart is added.

ABSTRACT

In this work, we describe and implement a framework for visualizing cluster computer metrics and metadata. Interaction methods for creating and modifying composite data charts are specified in conjunction with a dual MoireGraph interface.

Keywords: information visualization, cluster visualization

1 INTRODUCTION

Clusters of computers are used as a low-cost solution for handling large-scale computational tasks. While developing tasks for such systems, it is important to know the performance characteristics of the cluster so that sub-optimal properties can be found and addressed. Key areas that could be optimized include sources of bottlenecks, task loads, and scheduling. When working with clusters, there are a limited number of ways you can obtain information about the overall performance. A majority of these methods are textual in nature and produce such a large volume of heterogeneous data that it becomes impractical to analyze the data without some form of post processing. In order to practically analyze this information, a navigation scheme would be best.

Cluster computer visualizations are constructed in a hyperspace with at least four axes: cluster node, data item, data transform, and visual mapping. Typical cluster visualizations, such as Ganglia [1] and RockSoft's Cluster Top[3] only allow users to compare similar cluster metrics to one another. This work presents decomposes and organizes the hyperspace into a set of hyperplanes better suited to human comprehension.

2 DATA COLLECTION

Data was collected using Ganglia, a free and open source cluster metric collection and aggregation tool [1]. Ganglia collects a wide range of cluster metrics from any number of cluster nodes, including: CPU usage, memory usage, disk usage, etc. Our system builds upon the Ganglia data collection by organizing all of the metrics into files in a folder structure based on the cluster topology; this reports the overall statistics of the cluster. This topology includes the cluster summary information that Ganglia creates as the main node for the cluster. The metrics for each computer are stored in separate XML files. A metric listing was hand generated that contains the path, metric name, metric category, cluster node name, and the value range of the metrics. Each of the metric XML files is stored in a round robin archive format. This means that data for multiple time ranges is stored, with the data resolution inversely proportional to the length of the time period, resulting in some redundant data. In order to get a continuous series of data, CluVis reads in the metric XML, keeping only the highest resolution data for a given time sample. Each metric XML file is assigned to one DataSeries, and the DataSeries are pooled and shared for the entire application.

3 INTERFACE

We designed our interface in Python using the wxPython API for the windows and widgets. The OpenGL API was used for hardware rendering. The interface is divided into the visualization window and the data chart construction panel.

Department of Computer Science and Engineering, Bagley College of Engineering, Mississippi State University, MS 39762.

* crw7@msstate.edu

† tjk@acm.org

3.1 Visualization

The prevalent visual feature in CluVis is the usage of user-created DataCharts (Figure 2). DataCharts are one or more DataSeries rendered onto a texture. In our implementation, we chose to have each DataSeries represented by a polygon filled from the lower axis to the line generated from plotting the points from the DataSeries. Multiple DataSeries rendered on top of each other are blended additively, which allows the user to quickly identify similar and dissimilar regions of the data. Any DataChart with only one DataSeries will also render a labeled x/y axis. The names of the DataSereis' used to create the DataChart are rendered to the texture for identification. The texture is then cached and only regenerated when the DataSeries change.

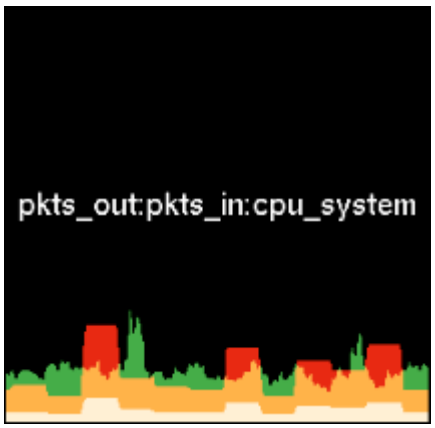

Figure 2. An example DataChart

The visualization contains two MoireGraph views: one for the cluster topology and the second for the current cluster data. The two MoireGraphs, focus+context graph views [2], are positioned one on top of the other. Each view has a white background and a black border in order to prevent confusion between the two views. When one view is in focus, it is transformed in such a way that it takes up more screen space than the view not in focus. The focused view is also the only view that is considered for the interaction methods defined by the MoireGraph. The user switches between the two views by clicking on the unfocused view with the mouse, which is followed by a smooth animated transition between the two focus states.

The topology view (Figure 1, left), positioned on the left side of the visualization, represents the topology of the cluster, with each node being a cluster computer. The nodes drawn in the topology view display the same metrics as the focus of the data view. The focus node on the topology view specifies the current domain of the visualization.

The data view (Figure 1, right), positioned on the right side of the visualization, displays one node for each set of metrics that have been associated into a data chart. The nodes displayed in the data view display the data from the domain specified by the focus of the topology view. Nodes that display metrics that are in a common category (CPU, Memory, Communication, Load, Disk) are connected. This means that nodes that display more than one category of metric are connected to two other nodes of both categories. Connecting the nodes in this way creates a wider range of connectivity in the underlying graph and allows the user to identify regions of interest.

The window also contains a chart preview for chart construction. This preview is positioned in the free space under the unfocused MoireGraph view. The chart preview reflects the current selections in the chart construction panel.

3.2 Chart Construction

The chart construction panel is used to specify which metrics to use in a data chart. In the panel, the metrics are represented by checkbox widgets. The checkboxes are grouped into categories, which are defined in the metric listing. There are also three buttons at the bottom of the panel: Add, delete, and clear. The add button adds a node to the data view with the metrics currently selected with the checkboxes. The delete button removes the current focus node from the data view. The clear button resets all of the checkboxes on the chart construction panel.

4 Discussion and Conclusions

CluVis allows cluster developers to analyze the performance of cluster machines to gain insight as to where and why performance bottlenecks are occurring. The data charts are rendered in such a way that user can find trends among different metrics, even if the metrics are of completely different categories. The dual graph layout allows the user to analyze multiple machines at the same time, which can help show if certain behavior is commonplace across the cluster. The chart construction panel allows the user to customize the visualization to their needs. These, combined with the basis of the graph connectivity, allow the user to explore the cluster's activity.

CluVis can be used as a component in cluster performance analysis. Other tools that drill down into specific performance measures and/or other diagnostics are needed to obtain a full picture of cluster utilization.

5 Future Work

Currently the chart construction only allows metrics to be displayed or not displayed; other properties could be associated with each metric. Future work could extend the chart maker to allow the user to customize how the charts are drawn. One such extension could be color pickers to choose which color to draw the individual metrics with. A history bar could be added to store snapshots of the two views. Opening the snapshot would load the saved views into the visualization.

References

[1] Ganglia, http://ganglia.sourceforge.net/. (current 28JUNE2005)

[2] T. J. Jankun-Kelly and Kwan-Liu Ma, "MoireGraphs: Radial Focus+Context Visualization and Interaction for Graphs with Visual Nodes," Proc. 2003 IEEE Symposium on Information Visualization, pp. 59–66, 2003.

[3] Rocks Cluster Distribution, http://www.rocksclusters.com/. (current 28JUNE2005)