Exploring Performance Data with Boxfish

Katherine Isaacs,1,3 Aadiya Landge2,3 Todd Gamblin1 Peer-Timo Bremer3 Valerio Pascucci2 Bernd Hamann1

1University of California, Davis 2SCI Institute, University of Utah 3Lawrence Livermore National Laboratory

Abstract

The growth in size and complexity of scaling applications and the systems on which they run pose challenges in analyzing and improving overall performance. To aid the process of exploration and understanding, we announce the initial release of Boxfish, an extensible tool for manipulating and visualizing data pertaining to application behavior. Combining and visually presenting data and knowledge from multiple domains, such as the application’s communication pattern and hardware’s network configuration and routing policies, can yield the insight necessary to discover the underlying causes of observed behavior. Boxfish allows users to query, filter and project across these domains to create interactive, linked visualizations.

Projecting, Filtering & Querying

Data can be projected from the domain in which it is generated onto another which might better explain its values [1]. Three domains and their native data are shown to the left. Projection is an element of Boxfish’s query and filter scheme. Data may be filtered by any attribute of any domain in which a projection exists before display.

Boxfish in Practice

Boxfish views have been used to aid in the understanding of node mappings, routing behavior and communication patterns [2, 3]. Boxfish views were used to show different node mappings [2]. Right – slow processes cluster in one plane [3].

Get Boxfish

Boxfish is licensed under [TBD]. Boxfish is written in Python 2.7 and runs on all major operating systems. To read full package requirements and download Boxfish, please visit [URL TBD].

Views in Boxfish

Boxfish comes with views of torus/mesh networks and nodes, communication graphs and data plots. New views can be added by taking advantage of Boxfish’s module API. We are actively developing new views.

References

Inspired by the data flow in Epinome [1], Boxfish allows filters and views to be grouped hierarchically. This allows both the data manipulation and the view interactions to be applied simultaneously to several modules at once.

When elements are selected in one view, corresponding elements in other views may be automatically highlighted depending on their position in the hierarchy and the policies of each subtree.

Though the elements shown in each view may not be the same, selection in one can induce selection in the others if the first set of elements can be projected onto the others.

Views in Boxfish

- **Plot**: Creates scatter plots and histograms from aggregated, filtered, and/or raw data. May be used to aid selection of extreme behavior.

- **Torus/Mesh 3D**: Displays data associated with the elements of a 3D torus or mesh network. Nodes and links are arranged in their natural configuration.

- **Torus/Mesh 2D**: Displays data associated with the elements of a 3D torus or mesh network. Nodes and links are arranged on a 2D plane to eliminate occlusion and ease selection.

- **Communication graph**: A vertex/edge representation of messages sent between processes.
Exploring Performance Data with Boxfish

Katherine Isaacs\textsuperscript{1,3} Aaditya Landge\textsuperscript{2,3} Todd Gamblin\textsuperscript{3} Peer-Timo Bremer\textsuperscript{3} Valerio Pascucci\textsuperscript{2} Bernd Hamann\textsuperscript{1}

\textsuperscript{1}University of California, Davis \textsuperscript{2}SCI Institute, University of Utah \textsuperscript{3}Lawrence Livermore National Laboratory

Boxfish in Practice

The 3D torus view and communication graphs aided in identifying the cause of a scaling problem in SAMRAI, an adaptive mesh refinement library.


Find out more about these, related visualizations and how they were applied on [Day of Week], in [Room] at [Time]!

The 2D torus projection has been used to understand network behavior and node mappings in pF3D, a laser-plasma interaction simulation.