Visual Analytics Techniques for Exploring the Design Space of Large-Scale High-Radix Networks

Jianping Kelvin Li*, Misbah Mubarak†, Robert B. Ross‡, Christopher D. Carothers§ and Kwan-Liu Ma*
*University of California, Davis, {kelli,klma}@ucdavis.edu
†Argonne National Laboratory, Lemont, IL 60539, USA, {mmubarak,rross}@anl.gov
‡Computer Science Department, Rensselaer Polytechnic Institute, chrisc@cs.rpi.edu

Abstract—**High-radix, low-diameter, hierarchical networks based on the Dragonfly topology are common picks for building next generation HPC systems. However, effective tools are lacking for analyzing the network performance and exploring the design choices for such emerging networks at scale. In this paper, we present visual analytics methods that couple data aggregation techniques with interactive visualizations for analyzing large-scale Dragonfly networks. We create an interactive visual analytics system based on these techniques. To facilitate effective analysis and exploration of network behaviors, our system provides intuitive, scalable visualizations that can be customized to show various traffic characteristics and correlate between different performance metrics. Using high-fidelity network simulation and HPC applications communication traces, we demonstrate the usefulness of our system with several case studies on exploring network behaviors at scale with different workloads, routing strategies, and job placement policies. Our simulations and visualizations provide valuable insights for mitigating network congestion and inter-job interference.**

Index Terms—**visualization, dragonfly networks, visual analytics, performance analysis**

I. Introduction

High-radix and low-diameter hierarchical network topologies such as the Dragonfly [1] have become popular choices for designing and building the next-generation high performance computing (HPC) systems. Many new and developing HPC systems, such as NERSC's Cori [2] and Argonne's Aurora [3] and Theta [4], are based on such network topologies.

Due to the increasing size and complexity of such networks, parallel discrete event-driven simulation (PDES) [5], [6] has emerged as a productive and cost-effective way to evaluate potential designs and explore different network configurations. However, the massive amount of data generated from large-scale network simulations create challenges to performance analysis and design space exploration. Much of the data generated by PDES is similar to data generated by a real HPC system, and thus will need to be analyzed in a similar manner. Pure numerical methods with automated analysis techniques, such as statistics and data mining, are insufficient for exploratory analysis of large and complex datasets [7], [8], so a visualization directed approach is a necessity. By combining visualization and data mining techniques to support exploratory analysis, visual analytics [9], [10] provides an effective mean to extract useful information from massive network performance data, providing insights for understanding complex network behaviors and improving system designs.

In this paper, we present a visual analytics system for exploratory analysis of large-scale Dragonfly networks. The design of our system has three primary goals with different challenges to be addressed.

Scalable Visualization. To analyze complex HPC interconnects with a large number of network links, routers, and compute nodes connected in a hierarchical structure, scalable visualizations are needed to depict and correlate various performance metrics and the structural properties at multiple granularities. Our visual analytics methods adapt data aggregation techniques to create scalable visualizations for summarizing the entire network with various levels of detail and showing the correlation between different performance metrics.

Flexible Exploration. Exploring the design space of Dragonfly networks requires domain expertise and background knowledge to steer the exploration process for studying different network behaviors, such as congestion and inter-job interference. Our visual analytics system allows customized visualizations to be flexibly created for the exploration task on hand, so that background knowledge can be easily applied to the exploration.

Simulation Integration. We aim to provide support for design space exploration using network simulations. Our system effectively processes and manages simulation data to provide not only interactive exploration but also quick comparison between simulation runs of different network configurations. Customized visualizations can be leveraged to effectively explore and compare selected performance metrics from different network configurations. We team up our visual analytics system with CODES [11], a high-fidelity network simulation toolkit, to explore network behaviors of large-scale Dragonfly networks. With effective visual analytics methods supporting the analysis of network simulations, the combined toolkits can be a powerful research vehicle for accelerating the design and development of HPC networks.

By analyzing network behaviors using our visual analytics system with CODES simulations, we demonstrate the advantages of our techniques and the usefulness of our system with several case studies that explore routing strategies, workload characteristics, job placement policies, and inter-job interference. By achieving our system design goals and using our system for exploring the design space of large-scale Dragonfly networks, we make the following contributions:

- We introduce visual analytics techniques to support exploratory analysis of large-scale HPC networks, and create an interactive analysis system based on these techniques to support design space exploration of Dragonfly networks.
- We develop methods and a user interface for rapidly specifying customized visualizations to explore various network behaviors and compare simulation results.
- We use our visual analytics system and network simulations to analyze representative workload from three parallel HPC applications. Our visualizations provide valuable information about the structural and temporal characteristics of each workload. Our analysis provides unique insights to the effect of adaptive routing and random job placement on inter-job interference and network congestion.
- Based on the insights from visual analysis of network behaviors, we develop a mitigation strategy that uses two different job placement policies with random allocation to reduce the effect of inter-job interference and improve application performance.

II. Background and Related Work

In this section, we first describe the Dragonfly network topology, and then we discuss related work in performance analysis of Dragonfly networks and performance analysis tools for HPC networks.

A. Dragonfly Topology

The Dragonfly network is a two-tier topology composed of g groups that are fully connected by all-to-all links. Each group has a routers, and each router has p terminals connected to it. Routers within a group are fully connected by local links. There are h global links in a router that are connected to the routers in other groups for inter-group traffic. A typical configuration for the Dragonfly topology [1] to achieve load balancing in network traffic is $a = 2p = 2h$, and the total number of groups is $g = a * h + 1$.

Multiple routing strategies are proposed for Dragonfly networks, including minimal, non-minimal, and adaptive routing [12], [13]. Contiguous job placement policy is typically used in supercomputer centers for allocating a consecutive set of compute nodes to each job. Studies [14], [15] show that various random job placement policies that randomly select a set of groups, routers, or individual terminals for each job can improve system performance.

B. Dragonfly Network Studies

Several studies have been conducted for analysis and evaluation of Dragonfly based networks. Mubarak et al. [16] used the ROSS [17] simulator to model large-scale Dragonfly networks and demonstrated scalable simulation performance on both the Blue Gene/P and Blue Gene/Q systems. Bhatele et al. [18] used BigSim to simulate a two-level direct network based on the Dragonfly topology and studied topology-aware mappings of different communication patterns. Jian et al. [14] presented a model to predict traffic of individual links on a Dragonfly network and provide analysis for different routing strategies and job placement policies. Yang et al. [15] used the CODES network simulation toolkit and application communication traces to study inter-job interference in a Dragonfly network with 1,056 nodes. They also showed that mixing contiguous and random node job placement policies can reduce the effect of job interference on application performance. Most of these studies are based on pure numerical and automated analysis, without employing visual analytics techniques. Our visual analytics methods are designed to complement existing numerical methods for studying Dragonfly networks, making analysis and exploration of large-scale networks more effective. As shown as part of previous studies [15], [19], inter-job interference and network congestion remain as important factors that impact network performance. Our visual analytics methods can help develop a better understanding for the effects of adaptive routing and random job placement policies on performance behaviors for large-scale Dragonfly networks.

C. Performance Analysis Tools

Many performance analysis tools can be used to analyze network performance data. Tools, such as ParaGraph [20], Jumpshot [21], and Vampir [22] use a visualization directed approach to analyze communication traces data. Most of these tools are based on the use of Gantt charts for visual analysis, so they do not scale well. Therefore, visual analytics techniques have been introduced to analyze massive MPI traces [23], [24], [25]. However, these works focused on the application domain and do not consider the structural properties of the physical network connections. Therefore, it is difficult to apply these techniques to exploring large networks due to the complexity in different network topologies. Some researchers have designed visualizations based on the topological properties of the HPC networks. Sigovan et al. [26] visualized a specialized I/O communication network in a radial node-link approach, while the communication patterns are summarized as 1D heatmaps along each edge, summarizing some properties such as latency, message size, etc. In this manner, consistent trends form rings around the network across multiple edges, while outliers stand out by themselves. While matching the visualization layout with the network topology, Landge et al. [27] visualized a 3D Torus network using a 3D view and multiple 2D projections to analyze the effect of MPI job mapping on network traffic. More recently, Bhatele et al. [19] modeled the behavior of the Dragonfly network using the Damselfly simulator and analyzed the effects of job placements and parallel workloads by visualizing all the routers and global channels between the routers in a radial layout. Most of these approaches visualized all the individual entities without using any data aggregation techniques. For large-scale hierarchical networks, direct visualization of the network topology does not scale, and thus scalability must be taken care of. In our work, we adapt hierarchical data aggregation and projection techniques to create high-level representation that preserves important structures and

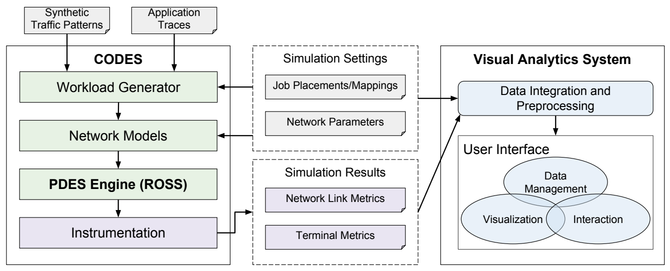


Fig. 1: Data flow for visualization and analysis of simulated networks.

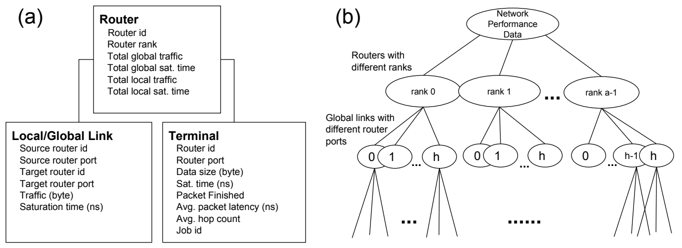


Fig. 2: (a) Entity tree is our basic data structure used for aggregating data with multiple entities. (b) An example of hierarchical data aggregation using the top-down approach [30] based on the topological properties of the Dragonfly network.

patterns in a large network. Moreover, existing tools usually use fixed visualization layouts for performance analysis and rarely provide flexibility for visualization customization. Fixed visualization layouts are usually designed for specific analysis tasks and thus cannot support general exploration of the design space of HPC networks. Our visual analytics system uses an extensible visualization layout with flexible customizations, allowing users to apply background knowledge to customize the visualization for different exploratory tasks. With effective visual analytics methods and customizable, scalable visualizations, our system supports effective analysis and exploration of large-scale network behaviors.

III. Simulation and Data Preprocessing

Our visual analytics system is designed to support the CODES network simulation toolkit [11] for exploring the design spaces of large-scale Dragonfly networks. CODES employs the Rensselaer Optimistic Simulator System (ROSS) [17], [28], a high-performance, parallel discrete-event simulator, to allow massive simulations to be run accurately at a packet level detail. The CODES simulation framework can simulate network workloads using synthetic traffic patterns. In addition, workloads to the simulated network can also be generated using parallel application traces from the DUMPI MPI trace library as part of SST macro toolkit [29].

For all the global links, local links, and terminals in the simulated networks, the following performance metrics are collected:

- Traffic on network links: The amount of data transferred in all the global, local and terminal links are recorded in the simulation.
- Link saturation time: The total time during which the buffers of the network virtual channels are full in the simulation.
- Average hop counts and packet latency: In addition to the traffic and saturation time for each terminal-router links, the simulation also records the average number of hops traversed and packet latency for each terminal.

To enable detailed exploration of network behaviors, we have extended the instrumentation capability in CODES to capture time series data for any given sampling rate, so we can visualize and analyze the temporal behaviors of workload characteristics and network performance. Users can select a certain phase or time range of the simulation for analysis and exploration. Figure 1 shows the data flow and coordination between CODES and our visual analytics system. From the inputs and outputs of CODES, we integrate and index all the data for effective visualization and analysis of large-scale Dragonfly networks. Network settings, such as job mapping information and connections of the network links, are retrieved from simulation configuration files. By integrating and preprocessing the simulation outputs with the network configuration files, our system can properly visualize the network connections and correlates performance metrics. Figure 2a shows the general data structure in our system for managing network performance data. The attributes of router id and router rank are used to reference data between routers to network links or terminals.

By closely coordinating with the CODES toolkit, simulation results can be immediately visualized and analyzed with our visual analytics system. Insights and new knowledge can be gained quickly with effective visual analytics, which can be used to change network configurations or refine the design choices in new simulation runs. Our system also provides effective visualizations for comparing simulation results between different network configurations, allowing users to better communicate and present the information and discoveries in the results.

IV. Visual Analytics Methods

We couple hierarchical data aggregation [30], [31], [32] with radial visualization layouts [33], [34], and extend both to support visual analysis of large-scale networks. By providing an interactive user interface for our visual analytics methods, our system allows users to create projection views of the aggregated data, which summarizes the characteristics of network performance. The projection views can be used with other visualizations to enable interactive exploration of various aspects of the network at different levels of granularity.

A. Data Aggregation

To create scalable visualizations for effective visual analysis, we aggregate the network performance data using techniques based on hierarchical clustering and binned aggregation. By considering the structures of the network topology, we iteratively cluster the data based on one of the network performance metrics or attributes to build an aggregate tree with multiple levels of detail about the network. Figure 2b shows an example of hierarchical aggregation on performance data of a Dragonfly network. At first, the performance data of the entire network is aggregated by the rank of the routers in a group, then it can be aggregated based on the link of the routers. To add another level of aggregation, we can further divide the global links into a histogram of six bins, for example, based on accumulated traffic of the link. For aggregate items, sum is used for most performance metrics, except the average value is used for the metric of average hop count and packet latency for the terminals. Because there are multiple entities (e.g., routers, terminals, and network links) in the network with each entity having different performance metrics (e.g., link saturation time and average packet latency), the entity tree (Figure 2a) is used as mappings for aggregating data and creating projection views. A detailed example is provided in Section IV-B2 after we presented our visualization techniques.

B. *Visualization Techniques*

To make use of the aggregate tree for analyzing and exploring large-scale network performance data, we use intuitive and scalable visualization to overview the entire network, analyzing different levels of details and correlating between various performance metrics in the network. Our visualization approach uses a radial layout, which is effective for depicting hierarchical data and providing useful insights about the information hierarchies [33]. Different performance metrics can be stacked in a hierarchical radial layout, providing overviews and correlations of hierarchical data. Lines or ribbons in the center of the radial layout can show the connections and communication patterns when visualizing interconnection networks, which helps for detecting network congestions and identifying bottlenecks. The high degree of symmetry in radial layouts can be helpful for checking traffic load balancing and comparing network behaviors.

1) Network Link Bundling: Our method also effectively visualizes a large number of network links as bundled connections for facilitating effective visual analysis and exploration. During data aggregation, we keep track of the local and global links between the routers and bundle the connections between the aggregate items. Two visual encodings (color and size) can be used for visualizing the traffic and saturation time of the network links, as shown in Figure 3. For example, if size is used to encode traffic and color is used to encode saturation time, the color of the ribbon represents the maximum saturation between the two ends. This visual mapping is effective for analyzing network congestions. Our visual encoding method for network links has an advantage over the matrix views, which are common visualizations used for performance and communication data. As our method can depict both traffic and saturation in one visual item, it is more effective for coordinating the network links and to see correlations between two performance metrics.

2) Visual Mapping: A key to create useful visualizations is selecting good visual mappings for the analysis tasks on hand.

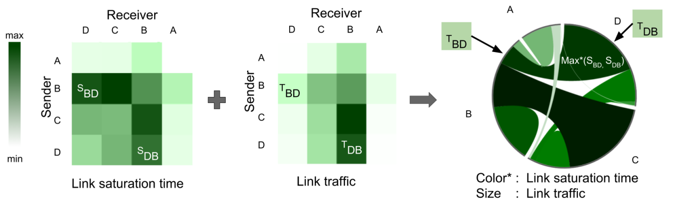


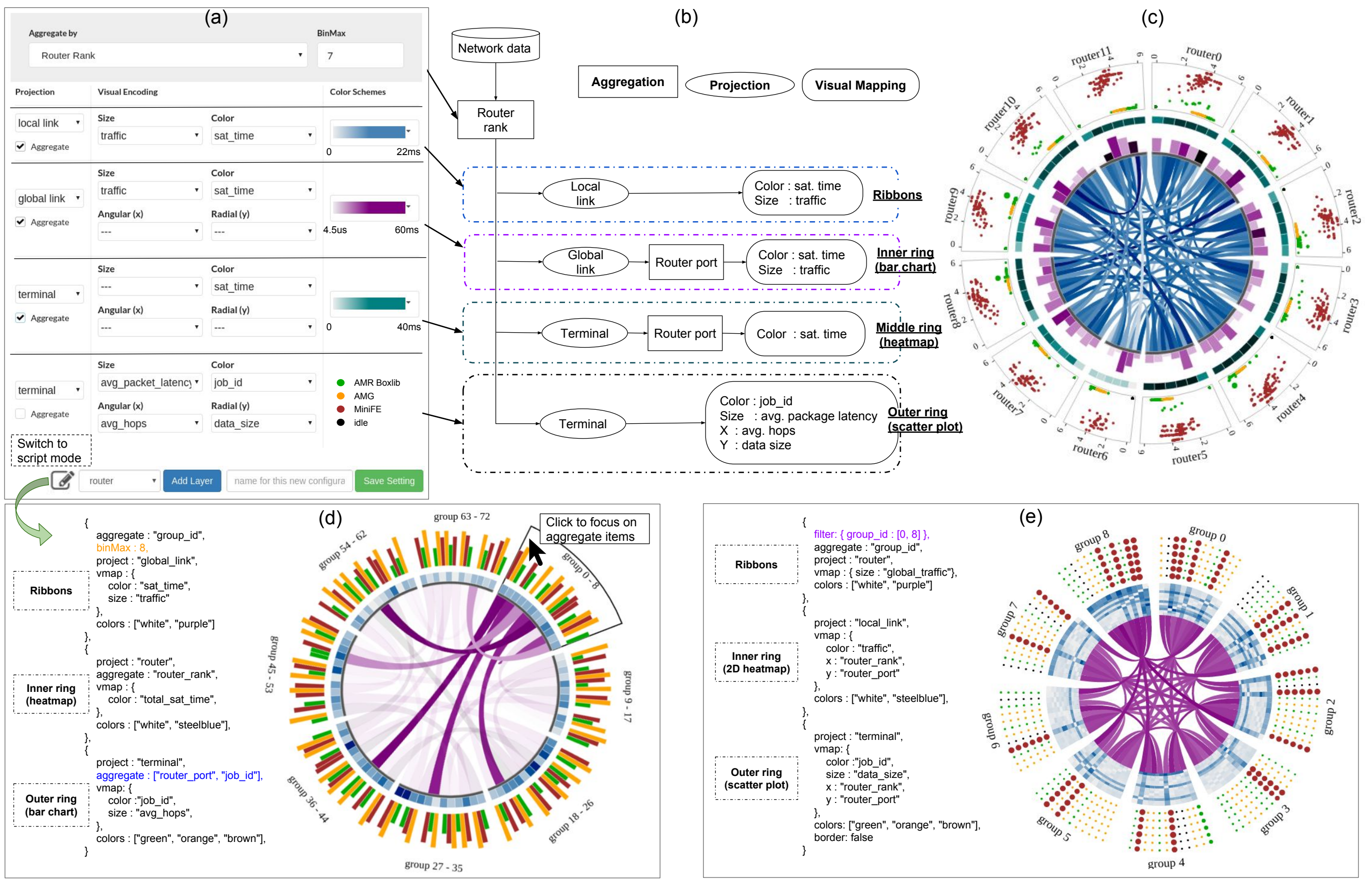
Fig. 3: Visual encoding for network links.

A visual mapping transforms data to visualization by mapping data attributes to the encodings (e.g., color and size) of visual items. While network performance data often has many more performance metrics than the number of visual encodings that we can use, the decisions for data aggregation and visual mapping require domain expertise and background knowledge. To allow users to easily steer the analysis and exploration by applying background knowledge, our system provides a visual interface (Figure 4a) for specifying data aggregation, projection, and visual mapping. The configuration is then used for data transformation (Figure 4b) to create the hierarchical radial visualization. When visualizing the aggregate tree using hierarchical radial visualizations, each ring in the radial layout represents one level of the tree, and each visual item on the ring is mapped to an aggregated performance metric. The result is the aggregated projection view of the whole network with multiple levels of detail, as shown in Figure 4c.

In the visual interface, users can select the performance metric or attribute for specifying the data aggregation. Users can control the number of layers in the visualization by adding a new level to the hierarchy or removing the existing ones. For each level, the user can select the entity and specify the visual mappings. Either the performance metrics of individual entities or their aggregate representations (toggling via the checkbox) can be visualized in the radial view. All performance metrics can be visualized as heatmaps, bar charts, or scatter plots in a circular fashion and stacked hierarchically to show correlations. The 1-D heatmap can be color coded for one metric. The bar chart can be visually encoded with the color and size. The 2-D heatmap can be encoded using color plus the radial and angular coordinates (or x and y for Cartesian coordinates). The scatter plot can have up to 4 visual encodings - color, size, x and y coordinates. The type of the plot used in each layer is based on the number of visual encodings defined by the user. A set of color schemes is provided for users to select at each level.

Our system also allows users to save the specification for the data transformation and visualization for analyzing another dataset or comparing between datasets. When comparing different datasets, the scale for visual encoding uses the same minimum and maximum values, which ensure fair comparison.

3) Projection View: As in the example given in Figure 4, the network performance data is first aggregated by the rank of the router within a group. The ribbons in the center of the radial visualization shows the intra-group connections between the routers with size representing the traffic and color repre-



Aggregate by: Router Rank
BinMax: 7

Projection	Visual Encoding		Color Schemes
local link ✓ Aggregate	Size: traffic	Color: sat_time	0 – 22ms
global link ✓ Aggregate	Size: traffic; Angular (x): ---	Color: sat_time; Radial (y): ---	4.5us – 60ms
terminal ✓ Aggregate	Size: ---; Angular (x): ---	Color: sat_time; Radial (y): ---	0 – 40ms
terminal Aggregate	Size: avg_packet_latency; Angular (x): avg_hops	Color: job_id; Radial (y): data_size	● AMR Boxlib ● AMG ● MiniFE ● idle

```
{
    aggregate : "group_id",
    binMax : 8,
    project : "global_link",
    vmap : {
        color : "sat_time",
        size : "traffic"
    },
    colors : ["white", "purple"]
},
{
    project : "router",
    aggregate : "router_rank",
    vmap : {
        color : "total_sat_time",
    },
    colors : ["white", "steelblue"],
},
{
    project : "terminal",
    aggregate : ["router_port", "job_id"],
    vmap: {
        color :"job_id",
        size : "avg_hops",
    },
    colors : ["green", "orange", "brown"],
}
```

```
{
    filter: { group_id : [0, 8] },
    aggregate : "group_id",
    project : "router",
    vmap : { size : "global_traffic"},
    colors : ["white", "purple"]
},
{
    project : "local_link",
    vmap : {
        color : "traffic",
        x : "router_rank",
        y : "router_port"
    },
    colors : ["white", "steelblue"],
},
{
    project : "terminal",
    vmap: {
        color :"job_id",
        size : "data_size",
        x : "router_rank",
        y : "router_port"
    },
    colors: ["green", "orange", "brown"],
    border: false
}
```

Fig. 4: Visual interface component (a) for specifying the data transformation and visual mapping (b) to create a hierarchical radial visualization (c), which is showing three jobs running on a Dragonfly network that has 73 groups with 12 routers and 72 terminals in each group. Scripts with high-level specifications can be used to create projection views with different levels of detail or different perspectives of the network, which can be used together to analyze network behaviors. The projection view in (d) shows a Dragonfly network with 73 groups aggregated to 8 partitions, which is running three jobs assigned using the random router job placement policy. The projection view in (e) shows the detail of the first 9 groups.

senting the saturation time (linearly interpolated from white to blue). On the inner ring, the global link entity is projected and aggregated by the router port and visualized as bar charts with color representing the saturation time and size representing the link traffic. On the middle ring, the terminal entity is projected and aggregated by the router port and visualized as 1-D heatmaps to show the aggregated saturation time. On the outer ring, the performance metrics of individual terminals are visualized as scatter plots. In the scatter plots, each dot represents a terminal with colors encoding the assigned job and sizes encoding the average packet latency. The angular (x) and radial (y) coordinates represent average hop counts and data size, respectively. In each scatter plot, the dots that are closer to the edge of the circle (toward the outside) represent terminals with higher values in average packet latency. The hierarchical radial visualization in Figure 4c reveals the typical intra-group communication patterns of the network and the correlations between the selected performance metrics as well as summarizing the performance characteristics.

Our system also allows the use of scripts for specifying projection views with more complex configurations. The script uses a key-value pair syntax (Figure 4d and 4e). Using scripts to specify the projection view is similar to using the visual interface but with more advanced configurations, such as filtering and using more than one attribute for aggregation (the blue and purple lines). The binMax parameter (yellow line) in Figure 4d limits the maximum number of bins for the aggregation. An extra binned aggregation is automatically performed if the result of data aggregation is greater than this limit. The resulting projection view shows a Dragonfly network with 73 groups aggregated to 8 partitions, with each partition containing multiple groups. Multiple projection views can be used together to show different levels of detail. Figure 4e shows a projection view with more details for the first 9 groups of the Dragonfly network. The filter operation in the script can be used to select a subset of the network for visualization.

C. Interactive Analysis and Exploration

Figure 5 shows the primary user interface of our visual analytics system. The customizable projection view can be used together with the other views to facilitate an effective

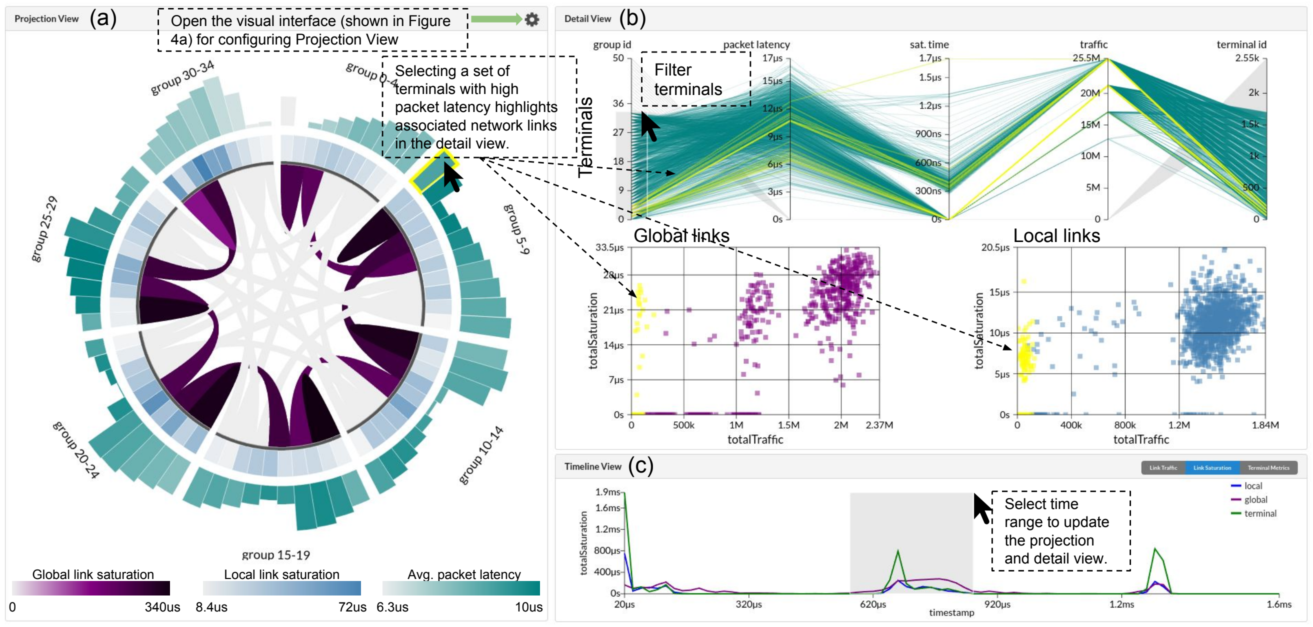


Fig. 5: The user interface of our visual analytics system consists the Projection view (a) for showing the aggregated performance metrics, Detail view (b) for visualizing individual network links and terminals, and Timeline view (c) for plotting the temporal statistics of the network links. The simulation result is for a Dragonfly network with 2,550 terminals running the AMG application with 1,728 MPI ranks.

process for interactive analysis and exploration of network performance. The detail view contains two scatter plots and a parallel coordinates plot [35]. The two scatter plots show the traffic and saturation of all the global and local links, while the parallel coordinates plot shows the performance metrics of all the terminals. The detail view thus provides a good context of the network performance, with useful background information that can help users decide how to configure the projection views for correlating performance metrics and revealing potential performance problems. The timeline view (Figure 5c) shows temporal statistics of either the total traffic/saturation for all type of network links, or the normalized mean values for the performance metrics of the terminals. A specific time range can be specified from the time line view to select the data to be shown in the projection and detail view. The detail view can also be used for filtering data or selecting subsets of the network. For interactive analysis and exploration, users can brush the axes of the parallel coordinate plot in the detail view for filtering data, and the projection views will be updated accordingly to represent the selected data. To examine details on demand, users can select a visual aggregate in the projection view to highlight (in yellow) the corresponding entities in the detail view. Such interactive exploration can reveal potential performance problems and bottlenecks.

V. Dragonfly Network Analysis

In this section, we use our visual analytic methods to study various factors that impact the performance of large-scale Dragonfly networks, and we compare and explore the design choices in different network configurations. In all our case studies, we use the Dragonfly network model proposed by Kim et al. [1]. We vary the scale of the network from 2,550 to 9,702 terminals. The performance metrics described in Section III are used for analyzing network performance. We use different synthetic workloads (nearest neighbor and uniform random traffic patterns) as well as workloads generated by communication traces from HPC applications.

A. *Communication Patterns*

As communication patterns can impact network performance, knowing the correlation of the communication pattern with the underlying network topology is helpful for identifying and avoiding bottlenecks. The projection view in our system can be used to reveal the communication patterns in a Dragonfly network, as well as correlating performance metrics to the communication pattern. Two examples are provided in Figure 6, which shows the results for two Dragonfly network simulations with 5,256 terminals running synthetic workloads using adaptive routing. The visualizations are created by aggregating the data by router rank. With the color of ribbons in the center showing the traffic on the local links, the concentric rings (from innermost to outermost) show the saturation time of the local links, global links, and terminals, respectively. For nearest neighbor workload, only one link between each pair of routers has traffic flow, as the terminals are only sending packets to their closest neighbor. Because of adaptive routing, we can also see some low traffic on other local links as they are used for non-minimal route in order to avoid congestion. For uniform random traffic, the terminals are randomly communicating with each other. Since this workload is balanced, the bundled links in the projection view have about the same amount of traffic, thus are shown with the same color.

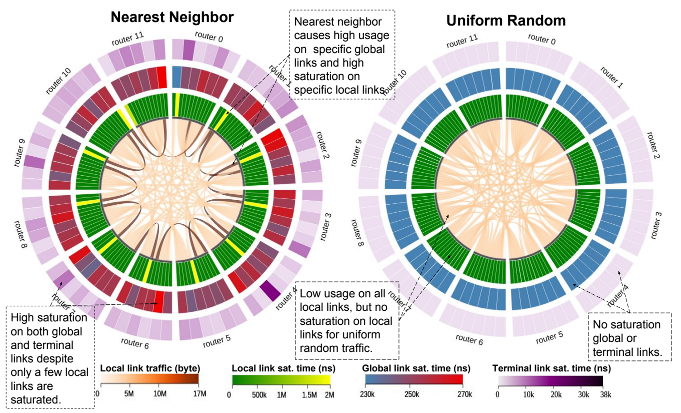


Fig. 6: Projection views showing intra-group communication patterns and the correlation between the saturation time of each type of network links on a Dragonfly network with 5,256 terminals using adaptive routing.

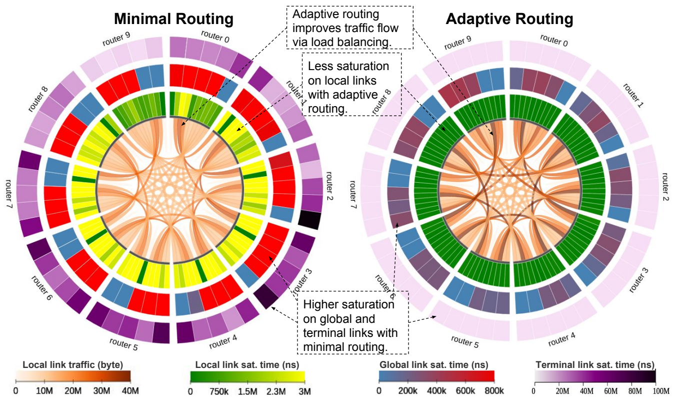


Fig. 7: Adaptive routing causes higher usage of local links and lower saturation time on all type of links on a Dragonfly network of 2,550 terminals running the AMG application.

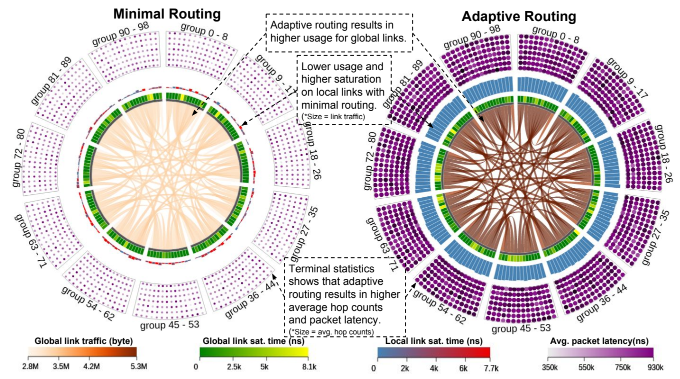


Fig. 8: Uniform random traffic on a Dragonfly with 9,702 terminals. Adaptive routing causes higher inter-group traffic, lower local link saturation time, and higher average hop counts and packet latency.

Using the same configuration for the projection view, Figure 7 shows a Dragonfly network of 2,550 terminals running the Algebraic Multigrid Solver (AMG) application with 1,728 MPI ranks. The AMG applciation has a 3D nearest neighbor communication pattern, where each terminal communicates with three other neighboring ranks instead of communicating to just one nearest neighbor. Therefore, more local links have higher traffic in AMG when comparing to the nearest neighbor traffic pattern.

In Figure 8, the projection views show the global link traffic in a Dragonfly network of 9,702 terminals running the uniform random workload, with the concentric rings (from inner to outer) depicting global link saturation time (color), local link traffic and saturation time (size and color), and terminal link saturation time, respectively. The projection view configurations in Figure 7 and Figure 8 can be used together to reveal both the inter-group and intra-group traffic patterns, allowing visual exploration of the correlation between the traffic patterns and the selected performance metrics. The correlation shown in circular hierarchies of the projection views helps us gain insights into the workload characteristics and network performance problems. Without the visual aggregation and summary provided in the projection views, it's difficult to see such correlations in a large-scale network.

B. Routing Strategies

An important factor that determines the performance of a Dragonfly network is the routing strategy for sending packets. To effectively compare routing strategies and network performance, projection views with the same configuration and visual encoding scales can be used. In Figure 7, we compare the network performance between minimal and adaptive routing strategies for the AMG application. As the colors of the ribbon encode the local link traffic and the colors of the rings encode saturation time of the local, global and terminal links, the figure clearly shows that adaptive routing results in higher intra-group traffic while having much lower saturation time for all type of network links when comparing to minimal routing. The network performance of minimal and adaptive routing for the uniform random workload is compared in Figure 8. By comparing the colors of the ribbons and the rings in the two projection views, it is clear that adaptive routing leads to higher usage of the global and local links than minimal routing. This is because adaptive routing randomly selects proxy groups for routing packets non-minimally to avoid network congestion, which doubles the bandwidth of the global links and leads to higher local link traffic in the proxy groups. Despite of higher traffic, the local links have no saturation with adaptive routing. On the other hand, minimal routing has a low utilization of local links but causes high saturation due to path conflicts. For average packet latency and hop count, the scatter plots in the projection views clearly show that adaptive routing results in higher hop counts and packet latency. As shown in this case study, our visual analytics approach allows fast and effective comparisons of different configurations for evaluating design choices of large-scale networks.

C. Application Performance Characterization

It is important to study the performance characteristics of Dragonfly networks using common workloads in parallel applications. Here we study and explore the network behaviors of the Dragonfly network using workloads captured from application traces which represent exascale workload behavior as part of the DOE Design Forward Program [36].

In particular, we analyze three of these communication traces:

- AMG: An Algebraic Multigrid Solver for unstructured mesh physics packages [37].
- AMR Boxlib: A single time step of an AMR run with compressible hydrodynamics and self-gravity [38].
- MiniFE: A Finite Element mini-application that uses a simple un-preconditioned conjugate-gradient algorithm to solve a sparse linear-system from the steady-state conduction equation [39].

Each of these applications has a different but representative communication pattern. Table I shows the information about the three applications. To analyze the application workload characteristics and the associated network performance behaviors, we run each application individually in a simulated Dragonfly network of 2,550 terminals. In all three simulations, we use adaptive routing and a contiguous job placement policy. Because the MPI ranks in each of the three applications are fewer than the number of terminals in the network, unused terminals are filtered out in the analysis.

Figure 9 shows the performance characteristics of the Dragonfly network with the three applications running individually. The projection views are using the same configuration as the one in Figure 4 except that the outermost ring is not used. They reveal the intra-group communication patterns with the correlations between the global link traffic/saturation and terminal link saturation for the three application workloads. The projection views in Figure 10 are based on the configuration used in Figure 4d, but the color encoding on the outer ring is changed to represent the average packet latency of the terminal since only one job is running in each of the simulations. Using the projection views in both Figures 9 and 10, we can effectively analyze the performance characteristics of each application workload. We can see that all three applications have high variances for the average package latency and hop counts, as shown in the outermost rings in the projection views. Both MiniFE and AMG have a balanced traffic across the local and global links. For AMG, most of the local links have the same level of saturation, with only a few of them having slightly higher saturation time. The global links connected to the first few groups apparently have much lower saturation

Application	Ranks	Data	Comm. Pattern
AMG	1728	1.2GB	3D nearest neighbor
AMR Boxlib	1728	2.2GB	Irregular and sparse
MiniFE	1152	147GB	Many-to-many

TABLE I: Summary of Applications.

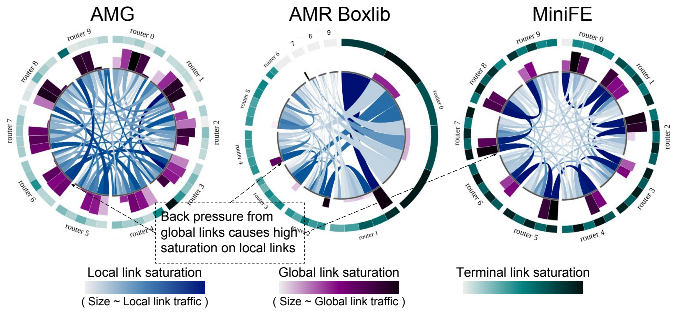


Fig. 9: Intra-group communication patterns and correlations between performance metrics of the local, global, and terminal links.

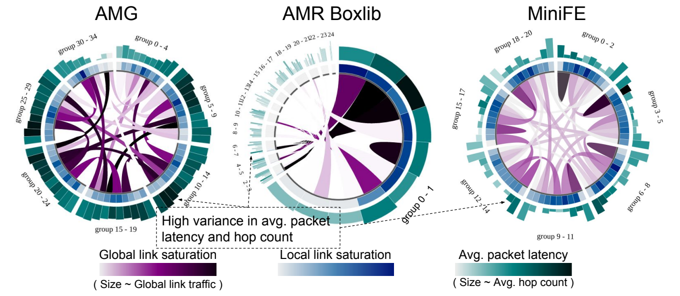


Fig. 10: Inter-group communication patterns and correlations between local link saturation and terminal performance metrics.

time than other global links. MiniFE has a many-to-many communication pattern, but the saturation time on most of the local links and global links is low, while only several local and global links have much higher saturation time. We can also see that the high saturation on local links is back pressure by the global links. On the other hand, AMR Boxlib has a very unbalanced load for both inter- and intra-group traffic, where the routers in the first two groups created more than 60 percent of the inter-group traffic, and the routers within the first 2 ranks contributed more than 50 percent of the intra-group traffic. In addition, these global and local links have high saturation time, while other links have very low or no saturation. If some of the traffic on these links can be redistributed to other links, the performance of AMR Boxlib would likely be improved.

In additional to the structural characteristics, we also analyze the temporal characteristics of the three application workloads and their effects on network performance. Figure 11 shows temporal characteristics of the three application workloads, with the timeline plots showing the total traffic over time for all types of network links. We can see that the three applications have very different temporal characteristics. An interesting observation is from the timeline plot for AMG, where it shows three traffic bursts in the beginning, middle, and near the end of the runtime. To further investigate the performance characteristics during the traffic bursts, we can analyze the link saturation over time, which is shown in Figure 5c. As we select the time range associated with the second traffic burst, the projection and detail views are showing the network performance behaviors of the selected time range.

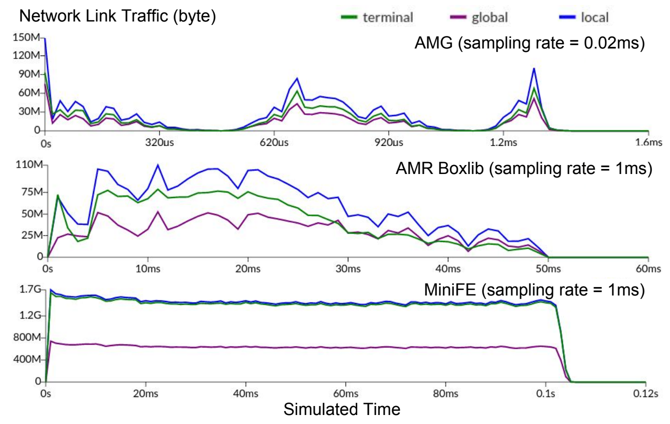


Fig. 11: Temporal characteristics of the network link traffic for three applications with different communication patterns.

From the projection view, we can see that only specific global links connecting the adjacent groups are highly saturated, while other global links have very low or no saturation. This points out that the standard adaptive routing algorithm [1] may be inefficient to avoid congestion during traffic bursts. As the network traffic burst occurs so quick that caused congestion in the minimal routes, the source router may not been notified immediately. In this case, a more advanced routing strategy should be used, such as progressive adaptive routing [12], where the decision to route non-minimally should be reevaluated every time before a non-minimal route is selected. In addition, interactive visualizations can be used to visually explore the network to search for abnormalities and bottlenecks, as shown in Figure 5a and 5b. When selecting a set of terminals with high packet latency and hop count in the outermost ring of the projection view, highlights in the detail views reveal that both the local links and global links connecting this set of terminals have high saturation time, which may lead to a performance bottleneck. If the congestion on either the local or global link is mitigated, we would be likely to see better packet latency for the terminals.

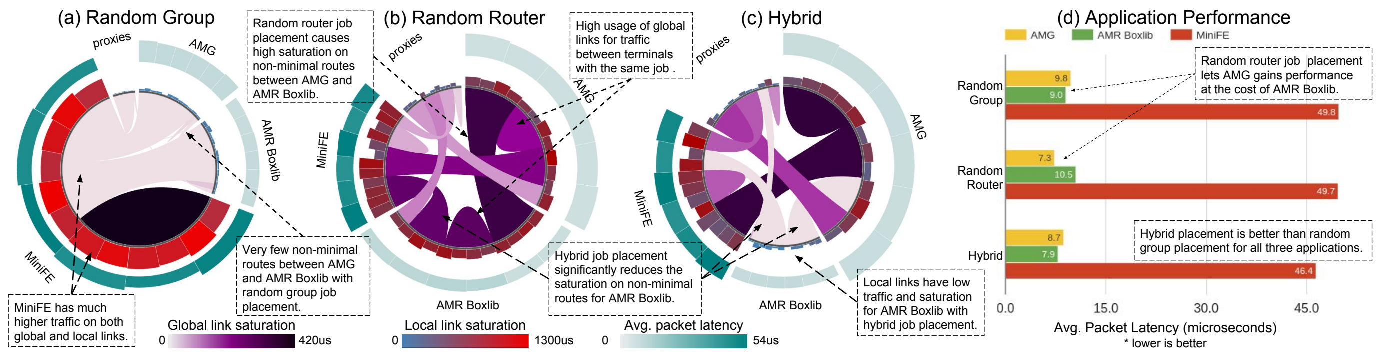
D. Inter-Job Interference

HPC systems usually run multiple jobs in parallel. Inter-job interference is another important issue in Dragonfly-based networks, which can significantly impact application performance. Job placement policy controls the allocation of available terminals to parallel applications (jobs), which is crucial for mitigating inter-job interference.

In this case study, we first analyze and compare inter-job interference for two job placement policies: random group and random router job placement. These two policies have been shown to bring performance gain as part of a previous study [14]. To explore and compare their effects on application performance, we simulate a Dragonfly network of 5,256 terminals running three application traces (AMG, AMR Boxlib, and MiniFE) in parallel. The network has a total of 73 groups, with each group having 12 routers and each router connecting to 6 terminals. The number of ranks and total data size of these three applications are listed in Table I. For random group placement, each job is assigned to a set of randomly selected groups in the Dragonfly network. Available terminals within the groups are assigned contiguously. For random router placement, each job is assigned to the terminals directly connected to a set of randomly selected routers. Figure 4e shows how the jobs are assigned to different routers in the first 9 groups of the network with the random router job placement policy.

Our visual analytics method provides an easy way for analyzing inter-job interference. Figure 12 shows the projection views for comparing the results of different job placement policies. By aggregating based on the job ID, the ribbons in the center of the projection views show global link traffic between the four different sets of groups or routers that are either assigned to one of the jobs (AMG, AMR Boxlib or MiniFE) or proxies, with size representing traffic and color representing saturation. The proxies are the routers with no terminal having job assigned, but being used for routing packets non-minimally in adaptive routing. In this case, the traffic on global links across different jobs is caused by the non-minimal routes. The size of the arcs in the projection view shows the ratios of the total traffic on global links for the routers associated with each job. Because the amount of global link traffic that a job sends to and receives from are the same, the two ends of each ribbon have equal size. The inner rings of the projection views show the aggregated performance metrics of the local links across router ranks, with color representing saturation and size representing traffic amount. The outer rings show the aggregated metrics for the terminals, with color representing average package latency and size representing average hop counts.

Figure 12a shows the simulation result with random group job placement. As MiniFE is much more communication-heavy than the other two jobs, it causes most of the traffic on global links, and it also has a much higher usage of local links within its assigned groups. Comparing to MiniFE, the traffic and saturation for both local and global links are very low for AMG and AMR Boxlib. For random router job placement, the projection view in Figure 12b shows that AMG and AMR Boxlib have higher usage of global links than MiniFE for both traffic among terminals with the same job and across terminals with different jobs. The global links associated with MiniFE have lower saturation time than the global links that are associated with AMG and AMR Boxlib. The routers assigned for AMG or AMR Boxlib also have much higher traffic and saturation on the local links when comparing to the random group job placement. We can also see that the global links between the routers assigned for AMG and AMR Boxlib have the highest saturation time. All these results are the impacts of inter-job interference and adaptive routing, where the communication-heavy job redistributes traffic to the routers assigned to other jobs. This causes the jobs with less communication to have high saturation on both local and global links. The application performance associated with the two job placement polices is shown in Figure 12d. When


Random router job placement causes high saturation on non-minimal routes between AMG and AMR Boxlib.

High usage of global links for traffic between terminals with the same job .

MiniFE has much higher traffic on both global and local links.

Very few non-minimal routes between AMG and AMR Boxlib with random group job placement.

Hybrid job placement significantly reduces the saturation on non-minimal routes for AMR Boxlib.

Local links have low traffic and saturation for AMR Boxlib with hybrid job placement.

Random router job placement lets AMG gains performance at the cost of AMR Boxlib.

Hybrid placement is better than random group placement for all three applications.

Fig. 12: Projection views (a,b,c) comparing three different job placement policies. The insights from our visualization and analysis of the random group and random router placements lead to the design of a new hybrid job placement policy that can mitigate inter-job interference and improve packet latency (d) for all three applications.

changed from random group to random router job placement, the performance of AMR Boxlib is degraded about 17%, as it has a sparse and irregular communication pattern and the least communication among the three jobs. However, MiniFE does not have noticeable gain in performance. This is because the high intra-group congestion for MiniFE prevents performance gain from adaptive routing. On the other hand, AMG has about 26% performance gain due to adaptive routing.

Our visualizations and analysis lead to better understanding of the coupled effect with adaptive routing and random job placement policies on network congestion and inter-job interference, which helps us gain insights to prevent performance degradation for AMR Boxlib. As shown by the visualizations, the minimal routes for AMR Boxlib are congested by other communication-heavy jobs in random router placement. To mitigate the inter-job interference that adversely impacts the performance of AMR Boxlib, we can use a hybrid job placement policy, where random router placement is used for MiniFE and AMG while random group placement is used for AMR Boxlib. This mitigation strategy can reduce inter-job interference for AMR Boxlib while allowing MiniFE and AMG to have performance gain with random router placement. Figure 12c shows the simulation result for the hybrid job placement. We can see that the traffic and saturation on both global and local links associated with AMR Boxlib are greatly reduced. As the performance results show in Figure 12d, the hybrid placement policy allows all three applications to have performance gain when comparing to the case with random group placement. AMR Boxlib has a 14% gain in performance instead of a 17% degradation, while AMG and MiniFE have 11% and 5% performance gain, respectively.

Here we simulate a Dragonfly network running a small number of jobs with different communication patterns for studying inter-job interference and evaluating different job placement policies. Our system allows us to effectively compare the network performance and quickly check the applicability of our optimization strategy. For cases with many more jobs running in a network, such as in production systems, we can select a subset of jobs for analysis. Alternatively, similar to the example in Figure 4d and 4e, we can group the jobs based on communication patterns or other classifications and use two projection views, with one view showing the impact of inter-job interference at a higher level and the other view analyzing the detail within the selected group of jobs. More importantly, this case study shows the effectiveness of our method for understanding complex network behaviors, identifying the causes of performance problems, and turning the insights from visual analysis into optimization.

VI. Conclusion

This paper presents visual analytics techniques and an interactive system for analyzing and exploring large-scale Dragonfly networks. Our techniques facilitating interactive analysis in user-defined novel spaces provide the needed flexibility to evaluate and explore the complex design choices of Dragonfly networks. As demonstrated with our case studies, our visualization and analysis can not only help develop better understandings of routing strategies, workload characteristics, and job placement policies, but also provide valuable insights for mitigating inter-job interference and network congestion, which can be used to improve overall system performance. Our case studies also show that simulation and visual analytics toolkits can be used together to support effective design space exploration of large-scale HPC networks, in which different network configurations can be rapidly simulated, evaluated, and explored to gain insights and knowledge for improving system designs. We believe our visual analytics techniques with flexible and scalable visualizations are widely applicable for performance analysis of HPC networks and systems. In future work, we plan to extend our system to support analysis and exploration of other network topologies, such as Fat Tree [40] and Slim Fly [41].

Acknowledgment

This research has been sponsored in part by the U.S. Department of Energy through grants DE-SC0007443, DE-SC0012610, and DE-SC0014917. Argonne National Laboratory's work was supported by the U.S. Department of Energy, Office of Science under contract DE-AC02-06CH11357.

References

[1] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-driven, highly-scalable dragonfly topology," in *ACM SIGARCH Computer Architecture News*, vol. 36, no. 3, 2008, pp. 77–88.

[2] N. J. Wright, S. S. Dosanjh, A. K. Andrew, K. B. Antypas, B. Draney, R. S. Canon, S. Cholia, C. S. Daley, K. M. Fagnan, R. A. Gerber *et al.*, "Cori: A pre-exascale supercomputer for big data and hpc applications," *Big Data and High Performance Computing*, vol. 26, p. 82, 2015.

[3] "Aurora argonne leadership computing facility," http://aurora.alcf.anl.gov/, 2016, accessed: 2017-04-10.

[4] "Theta argonne leadership computing facility," https://www.alcf.anl.gov/theta, 2016, accessed: 2017-04-10.

[5] R. M. Fujimoto, "Parallel discrete event simulation," *Communications of the ACM*, vol. 33, no. 10, pp. 30–53, 1990.

[6] ——, *Parallel and distributed simulation systems*. Wiley New York, 2000, vol. 300.

[7] D. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler, "Visual analytics: Scope and challenges," *Visual data mining*, pp. 76–90, 2008.

[8] D. A. Keim, F. Mansmann, and J. Thomas, "Visual analytics: how much visualization and how much analytics?" *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 2, pp. 5–8, 2010.

[9] K. A. Cook and J. J. Thomas, "Illuminating the path: The research and development agenda for visual analytics," Pacific Northwest National Laboratory (PNNL), Richland, WA (US), Tech. Rep., 2005.

[10] J. J. Thomas and K. A. Cook, "A visual analytics agenda," *IEEE computer graphics and applications*, vol. 26, no. 1, pp. 10–13, 2006.

[11] M. Mubarak, C. D. Carothers, R. B. Ross, and P. Carns, "Enabling parallel simulation of large-scale hpc network systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. PP, no. 99, pp. 1–1, 2016.

[12] N. Jiang, J. Kim, and W. J. Dally, "Indirect adaptive routing on large scale interconnection networks," in *ACM SIGARCH Computer Architecture News*, vol. 37, no. 3. ACM, 2009, pp. 220–231.

[13] J. Won, G. Kim, J. Kim, T. Jiang, M. Parker, and S. Scott, "Overcoming far-end congestion in large-scale networks," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2015, pp. 415–427.

[14] N. Jain, A. Bhatele, X. Ni, N. J. Wright, and L. V. Kale, "Maximizing throughput on a dragonfly network," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Press, 2014, pp. 336–347.

[15] X. Yang, J. Jenkins, M. Mubarak, R. B. Ross, and Z. Lan, "Watch out for the bully! job interference study on dragonfly network," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, ser. SC*, vol. 16, 2016.

[16] M. Mubarak, C. D. Carothers, R. Ross, and P. Carns, "Modeling a million-node dragonfly network using massively parallel discrete-event simulation," in *High Performance Computing, Networking, Storage and Analysis (SC), 2012 SC Companion:*, 2012, pp. 366–376.

[17] C. D. Carothers, D. Bauer, and S. Pearce, "Ross: A high-performance, low-memory, modular time warp system," *Journal of Parallel and Distributed Computing*, vol. 62, no. 11, pp. 1648–1669, 2002.

[18] A. Bhatele, N. Jain, W. D. Gropp, and L. V. Kale, "Avoiding hot-spots on two-level direct networks," in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2011, p. 76.

[19] A. Bhatele, N. Jain, Y. Livnat, V. Pascucci, and P.-T. Bremer, "Analyzing network health and congestion in dragonfly-based supercomputers," in *IEEE International Parallel and Distributed Processing Symposium*, 2016, pp. 93–102.

[20] M. T. Heath and J. E. Finger, "Paragraph: A tool for visualizing performance of parallel programs," in *Second Workshop on Environments and Tools for Parallel Sci. Comput*, 1994, pp. 221–230.

[21] O. Zaki, E. Lusk, W. Gropp, and D. Swider, "Toward scalable performance visualization with Jumpshot," *Inernational Journal of High Performance Computing Applications*, vol. 13, pp. 277–288, Aug. 1999.

[22] W. E. Nagel, A. Arnold, M. Weber, H.-C. Hoppe, and K. Solchenbach, "Vampir: Visualization and analysis of MPI resources," *Supercomputer*, vol. 12, pp. 69–80, 1996.

[23] C. Muelder, F. Gygi, and K.-L. Ma, "Visual analysis of inter-process communication for large-scale parallel computing," *IEEE Trans. on Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1129 –1136, Nov.-Dec. 2009.

[24] K. E. Isaacs, P.-T. Bremer, I. Jusufi, T. Gamblin, A. Bhatele, M. Schulz, and B. Hamann, "Combing the communication hairball: Visualizing parallel execution traces using logical time," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2349–2358, 2014.

[25] C. Sigovan, C. Muelder, and K.-L. Ma, "Visualizing large-scale parallel communication traces using a particle animation technique." *Comput. Graph. Forum*, vol. 32, no. 3, pp. 141–150, 2013.

[26] C. Sigovan, C. Muelder, K. Ma, J. Cope, K. Iskra, and R. B. Ross, "A visual network analysis method for large scale parallel i/o systems," in *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2012, pp. 308–319.

[27] A. G. Landge, J. A. Levine, A. Bhatele, K. E. Isaacs, T. Gamblin, M. Schulz, S. H. Langer, P.-T. Bremer, and V. Pascucci, "Visualizing network traffic to understand the performance of massively parallel simulations," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 12, pp. 2467–2476, 2012.

[28] P. D. Barnes Jr, C. D. Carothers, D. R. Jefferson, and J. M. LaPre, "Warp speed: executing time warp on 1,966,080 cores," in *Proceedings of the 1st ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 2013, pp. 327–336.

[29] A. F. Rodrigues, K. S. Hemmert, B. W. Barrett, C. Kersey, R. Oldfield, M. Weston, R. Risen, J. Cook, P. Rosenfeld, E. CooperBalls *et al.*, "The structural simulation toolkit," *ACM SIGMETRICS Performance Evaluation Review*, vol. 38, no. 4, pp. 37–42, 2011.

[30] N. Elmqvist and J.-D. Fekete, "Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 3, pp. 439–454, 2010.

[31] M. C. Hao, U. Dayal, R. K. Sharma, D. A. Keim, and H. Janetzko, "Visual analytics of large multidimensional data using variable binned scatter plots," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2010.

[32] Z. Liu, B. Jiang, and J. Heer, "immens: Real-time visual querying of big data," in *Computer Graphics Forum*, vol. 32, no. 3pt4. Wiley Online Library, 2013, pp. 421–430.

[33] J. Stasko and E. Zhang, "Focus+ context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations," in *Proceeding of InfoVis*, 2000, pp. 57–65.

[34] C. Sigovan, C. W. Muelder, and K.-L. Ma, "Visualizing large-scale parallel communication traces using a particle animation technique," in *Computer Graphics Forum*, vol. 32, no. 3pt2. Wiley Online Library, 2013, pp. 141–150.

[35] A. Inselberg and B. Dimsdale, "Parallel coordinates," in *Human-Machine Interactive Systems*. Springer, 1991, pp. 199–233.

[36] "Characterization of the DOE mini-apps," http://portal.nersc.gov/project/CAL/doe-miniapps.htm, 2016, accessed: 2016-06-30.

[37] U. M. Yang *et al.*, "Boomeramg: a parallel algebraic multigrid solver and preconditioner," *Applied Numerical Mathematics*, vol. 41, no. 1, pp. 155–177, 2002.

[38] J. Bell, A. Almgren, V. Beckner, M. Day, M. Lijewski, A. Nonaka, and W. Zhang, "Boxlib users guide," *github. com/BoxLib-Codes/BoxLib*, 2012.

[39] G. Nikishkov, "Introduction to the finite element method," *University of Aizu*, 2004.

[40] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, 2008, pp. 63–74.

[41] M. Besta and T. Hoefler, "Slim fly: a cost effective low-diameter network topology," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Press, 2014, pp. 348–359.