Parallel Visualization for GIS Applications

Alexandre Sorokine, Jamison Daniel, Cheng Liu

Oak Ridge National Laboratory, Geographic Information Science & Technology,
PO Box 2008 MS 6017, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6017

Introduction

In the recent decade GIS researchers have experienced enormous growth of the size and complexity of the geographic data. Among the factors that have contributed to such growth are the introduction of easily available high-resolution satellite imagery, the development of new acquisition methods for remotely-sensed data like LIDAR, the availability of a large number of geographic datasets through the Internet that can be integrated together, and the advancement of large-scale simulation models - including those that utilize parallel computing architectures.

Traditionally, the growth of computational resources available to researchers was on par with the growth of data size. CPU speed, amount of RAM, and disk space available on an average desktop PC have increased orders of magnitude just within a few years. The same is true of the performance graphical processors. However, display size and display resolution did not grow as fast as other characteristics of a personal computer. This can be explained by both hardware and software limitation of the existing display technology and by the lack of demand for higher display resolutions from the users of standard office applications.

The use of large high-resolution displays was confined to the areas of engineering and scientific visualization where small screen size and low display resolution were significant limiting factors for comprehension of large data sets. At the same time availability of specialized visualization equipment was limited only to organizations that could afford systems costing hundreds of thousand and millions of dollars.

Newly emerged solutions for large-screen high-resolution visualization employ clusters of commodity personal computers. Such clusters allow creation of very large tiled displays while keeping acquisition and maintenance costs significantly lower than for traditional high-end visualization systems. This paper discusses the theory and practical issues pertaining to the use of such clusters for geographic visualization.

The paper is structured as follows. Section 1 presents an architectural overview of parallel visualization systems that are built around commodity hardware and a short summary of the available software for parallel visualization. Section 2 discusses the problem of parallel visualization in the GIS context. Section 3 presents an approach used at ORNL for parallel GIS visualization and simulation that is followed by conclusions and an outline of the directions for future research in Section 4.

Current Developments

Hardware Architecture

Larger number of pixels and larger display size can be achieved by combining several smaller display units (LCD or CRT screens or projectors) to produce a single tiled display. The system has to be set up so that the updates of all display units are synchronized and each unit receives a proper portion of the whole image.

One of the approaches to create such a system is to use several networked PCs each equipped with a monitor and set up to work in parallel. Typical architecture of a parallel visualization system is shown on Fig. 1. It consists of a number of rendering nodes (PCs) with display devices connected to them and a head node that distributes data and workload between the rendering nodes. Each node may have one or several CPUs, a hard drive, RAM and one or more graphics cards and a corresponding number of display devices connected to it.

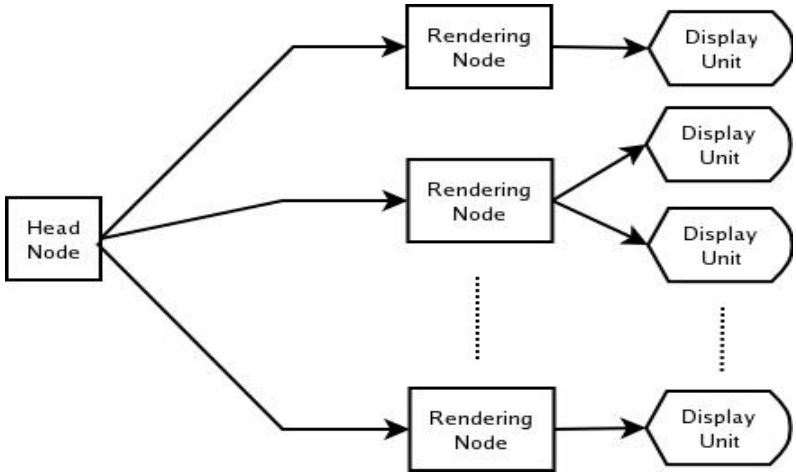

Figure 1.

An example of a such system at ORNL is the EVEREST visualization cluster. This system combines thirty-two commodity dual processor Opteron PCs running the Linux OS. Twenty-seven back projected 3500 lumen DLP projectors, each displaying a native resolution of 1280x1024, are hard edge aligned to create an 11,520x3,072 contiguous pixel array that physically stretches 30 feet horizontally and rises more than 8 feet vertically [Daniel, 2004].

Software Architecture and Applications for Parallel Visualization

The goal for building of an optimal architecture for parallel visualization is to efficiently distribute workload between various elements of the visualization system. The data being visualized first has to be read from a disk, then it is processed by a CPU and final rendering is performed by the graphics card processor.

Several architectures of a distributed rendering pipeline have been described in literature [Bethel et al., 2003]. In this study we investigate applicability of the “sort-first” and “sort-last”

architectures. In the "sort-first" technique, distribution of graphics primitives between rendering nodes is performed in the object coordinates prior to rendering. Rendering and rasterization is performed on separate nodes. In the "sort-last" technique only rasterized pixels are distributed between the nodes. Both of these techniques have certain advantages and disadvantages. "Sort-last" allows to have predictable communication patterns between nodes and does not require modification of standard X/Windows applications. However, "sort-last" can result in significant performance penalties because most of the graphic operations are not parallelized and have to be performed on a single node. "Sort-first" technique allows more fair distribution of load between the nodes.

Several Linux system software packages can support various techniques for parallel visualization and are used in this project. Xdmx (http://dmx.sf.net/) is a distributed X server that implements the "sort-last" approach. Chromium (http://chromium.sf.net/) is a system for distributed interactive rendering of OpenGL applications which is able to implement the "sort-first" technique.

In the domain of the application software several systems are available that support parallel visualization. Examples of such packages are VisIt (http://www.llnl.gov/visit/) and OpenDX (http://opendx.org/). The main areas of application of these software systems is scientific and engineering visualization like, for example, three-dimensional visualization of CT scans or visual representation of very large datasets resulting from physical simulations.

GIS in the Context of Parallel Visualization

Almost any GIS visualization tool capable of running in the X/Windows environment can be distributed using the "sort-last" approach offered by Xdmx. However, in the case of GIS, a strict sort-last distribution model leads to a significant performance disadvantage. The data must be rendered to a local X-server configured to the full resolution of the distributed display array. This poor exploitation of the available system resources restricts the display nodes to performing as simple networked framebuffers while the headnode is burdened with all geometric operations and full framebuffer maintenance. To more fully exploit the hardware a hybrid approach using both sort-last composition and sort-first geometric distribution is necessary. Geometric primitives can be sorted intelligently on the cluster headnode and distributed to the appropriate display nodes for further graphical computations. As a result, the headnode can spend more of its resources on sort-last composition and an aggregate performance advantage is achieved.

At the moment of writing of this paper the authors were not aware of any visualization tools capable of performing parallel rendering while utilizing GIS problem-specific knowledge. The use of visualization software that supports "sort-first" (like VisIt and OpenDX) is limited due to the lack of features needed for practical visualization of GIS data. The most important limiting factor is the lack of support for cartographic projections and coordinate systems that prevents integration of various GIS datasets on the same display. Other missing features are:

- support for cartographic symbolization,
- ability to work with or import GIS data formats,

? ability to show multiple layers of data at once,

? support for GIS-specific data models such as categorical rasters or topological vector data.

This problem can be solved by either adding GIS capabilities to visualization systems or by adapting existing GIS software for parallel visualization.

Approach of this Study

In this study the EVEREST cluster is used to implement an evacuation planning model utilizing both visualization and computational parallelism. The purpose of the model is to simulate evacuation of population from an area affected by an emergency. An emergency is an unforeseen situation that threatens people such as floods, hurricanes, tornadoes, fires, toxic gas releases, chemical spills, radiological accidents and explosions. The model can be used to test various evacuation plans before actual emergency happens.

The objective of this model is to minimize the total time needed for evacuation of people from a congested and capacitated network. The input to this model consists of an existing transportation network with attributes of link capacity and travel time, current traffic flow from road sensors [Shankar et al., 2005], and a high-resolution population database [Bhaduri et al., 2002]. The model generates an evacuation plan with individual trips, the trip's starting time and it's exiting the emergency area time. This evacuation plan is visualized dynamically on the display of the cluster. The model is based on the constrained shortest path algorithm that is an extension of the shortest path algorithm capable to account for route scheduling with capacity constraints [Lu and Shekhar, 2004].

Handling of large data sets is achieved by the parallelization of the algorithms by means of spatial decomposition. The network is decomposed into major highways and local streets. The local streets are decomposed further to smaller geographic regions. A load balancing algorithm distributes the sub-networks to each node in the cluster. The model is programmed in C/C++ and uses MPI standard for inter-node communication.

At the beginning of a model run the chunks of network data are distributed across the cluster nodes. Each simulation step is solved using the CPU of a particular node. The results of each step are aggregated by the head node and redistributed back to the nodes for rendering and visualization on the corresponding display tile.

Conclusions

Inexpensive Linux-based visualization clusters can strongly benefit the area of geographic visualization and especially visualization of large geographic data sets.

There is no doubt that GIS parallel visualization is still in its nascence. Effective GIS visualization software capable of fully utilizing the power of parallel visualization systems is yet to be developed. Also there is a need of more basic research in such areas as:

? new cartographic methods that are adequate for very high-resolution large dynamic displays,

? parallelization algorithms that employ knowledge specific to GIS domain.

References

[Bethel et al., 2003] Bethel, E.W., Humphreys, G., Paul, B., and Brederson, J. D. (2003). Sort-first, distributed memory parallel visualization and rendering.In *IEEE Symposium on Parallel and Large-Data Visualization and Graphics*, pages 41 50. IEEE.

[Bhaduri et al., 2002] Bhaduri, B., Bright, E., Coleman, P., and Dobson, J. (2002). Landscan: Locating people is what matters. *Geoinformatics*, 5(2):34 37.

[Daniel, 2004] Daniel, J. R. (2004). A middleware graphical toolkit for multiframed display environments. Master s thesis, The University of Tennessee, Knoxville.

[Lu and Shekhar, 2004] Lu, Q. and Shekhar, S. (2004). Capacity contrained routing for evacuation planning. In *Intelligent Transportation Systems Safety and Security Conference*, Miami.

[Shankar et al., 2005] Shankar, M., Smith, C., and Gorman, B. (2005). The sensornet node: Last-mile platform for interoperability. In *Information Processing in Sensor Networks.*