Three Dimensional Animation for Performance Debugging Utilizing Human Cognitive Ability

Noritaka OSAWA and Toshitsugu YUBA

The University of Electro-Communications
1-5-1 Chofugaoka, Chofu-shi, Tokyo, JAPAN
{osawa,yuba}@is.uec.ac.jp

ABSTRACT Visualization is a useful first step for performance debugging, that is, performance tuning of a parallel program if it shows the balance between computation and communication. In visualizing performance data, it is important to exploit the human cognitive ability of recognizing slight differences in shapes, positions and velocities of familiar objects. We have proposed animation for performance debugging that utilizes this human cognitive ability. It animates performance data based on dynamic system modeling where the data is mapped to a dynamic system model, the model is simulated and the result of simulation is made visible in 3 dimensions.

KEYWORDS 3-D animation, performance debugging, parallel program, dynamic system, cognitive ability

1. INTRODUCTION

In parallel programming, not only are parallel algorithms very important but also distribution of data and computation because inappropriate processor scheduling and data placement make a parallel program slower. It is important to schedule distribution of computation and data appropriately in addition to using a suitable parallel algorithm in parallel programming. Generally speaking, performance debugging (performance tuning) is needed in addition to logical debugging.

The first step of performance debugging is to find locations of bad balance (performance bugs). Visualization is useful to help find performance bugs. Visualization of behavior and performance of parallel programs have been studied (Heath, 1995). They directly map state values or their statistical values, which represent states of a parallel program or a parallel computer, to *graphical attributes* like coordinates, shapes, color and brightness.

We propose animation of parallel processing states that exploits human cognitive ability (OSAWA, 1996). Our proposed animation does not directly map the states into graphical attributes but maps them to bodies and forces in a dynamic system model that is familiar to human beings.

2. ANIMATION BASED ON DYNAMIC SYSTEM MODELING

A goal of visualization of data is to present data in a form that a human being is familiar with and is easily understood. Utilizing human cognitive ability, the user can understand states that are represented by a lot of values.

A human being lives in a dynamic world macroscopically and is familiar with motion in a dynamic system. Thus we have considered utilizing or exploiting the human ability to easily recognize balanced states, unbalanced states and natural (unforced) movement in a dynamic system. We propose an animation method of state values where we map the values into quantities of states in a dynamic system model, then simulate the dynamic system, and

Human–Computer Interaction: INTERACT'97 S. Howard, J. Hammond & G. Lindgaard (editors)
Published by Chapman & Hall

make the result of simulation visible. This animation shows changes in balance between computation and communication as movement in a familiar dynamic system.

First, we need to determine the dynamic system to be used, and the mapping of elements and structures in a parallel computer to bodies and forces in the dynamic system. We can model a parallel computer based on its network topology. For example, processor elements (PEs), computational load of a PE, communication links and amounts of communication in a parallel computer can be mapped to bodies, mass of a body, springs among bodies and attractive forces among bodies respectively. Mapping to a dynamic system does not necessarily refer to precise mapping to the real world.

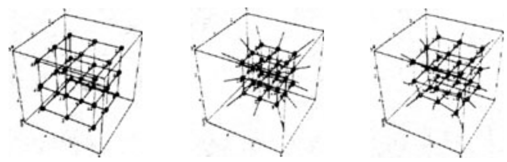
Next, we formulate equations of motion to simulate the dynamic system model. The motion in the system is simulated by solving the equations of motion (that is, an initial value problem of ordinary differential equations) numerically.

The simulation result can be visualized using usual visualization techniques. Not only visualization techniques for performance debugging but also techniques of scientific visualization can be used.

Let us consider a parallel computer that takes 3 x 3 x 3 computational nodes and communication links in 3-D mesh topology as shown in Figure 1 (a). Elastic force is exerted between neighbor nodes to maintain the structure. Communication between nodes applies attractive force to them. Weak repulsive forces are exerted among bodies. Frictional force is caused that is proportional to the velocity of a body.

Figure 1 shows snapshots of animation based on the above modeling and pseudo trace data. In Figure 1, a small cube and a thin cylinder represent a processor element and a communication link respectively. Whiskers represent tracks of movement of the cubes. Figure 1(a) shows the initial state. When communication load is balanced, the shapes of 3-D mesh structure is almost maintained as shown in Figure 1(b). We can see that the communication is imbalanced by looking at the distortion of the structure in Figure 1(c).

Spring model (Kamada, 1989) and other models have been studied in static planar graph drawing. They are similar to our method in relation to visualization by use of a dynamic model. Their drawings are based on a stable solution of the system however our animation is based on solving equations of motion. Furthermore, our proposed method changes external force and mass of objects depending on trace data of message transfer and computational load on nodes.



(a) Initial (b) Balanced (c) Imbalanced

Figure 1: Snapshots of Animation Based on Pseudo Trace Data on a Parallel Computer with 3-D Mesh Topology Network.

3. CONCLUDING REMARKS

This paper proposed 3-D animation for performance debugging of a parallel program that utilizes the human cognitive ability, and showed snapshots of animation examples for performance debugging and effectiveness of our proposed animation. Animation of the system is based on dynamic system modeling, simulation and visualization.

4. REFERENCES

Heath, Michael T., Allen D. Malony and Diane T. Rover (1995) Parallel Performance Visualization: From Practice to Theory. *IEEE Parallel & Distributed Technology*, Vol. 3, No. 4, pp. 44-60.

Kamada, T., and S. Kawai (1989) Algorithms for drawing general undirected graphs. *Information Processing Letters*, Vol.31, No.1, pp.7-15.

Osawa, Noritaka and Toshitsugu Yuba (1996) Animation of Parallel Computer Behavior Based on Dynamic System Modeling. *Fourth Workshop on Interactive Systems and Software*, pp.189-197, 1996 (in Japanese).