MPE Graphics—Scalable X11 Graphics in MPI

William Gropp Edward Karrels
Ewing Lusk
Mathematics and Computer Science Division
Argonne National Laboratory
(Abstract)

Introduction. As parallel programs enter the mainstream, they need to provide the same facilities and ease-of-use features expected of uniprocessor programs. For many applications, this means that they need to provide graphical output. This talk discusses a library of routines that provide scalable X Window System graphics. These routines make use of the MPI message-passing standard to provide a safe and reliable system that can be easily used in parallel programs. At the same time they encapsulate commonly-used services to provide a convenient interface to X graphics facilities.

The easiest way to provide X11 graphics to a parallel program is to allow each process to draw on the same X11 Window. That is, each process opens a connection to the X11 server and draws directly to it. In one sense, this is as scalable a system as possible, since the single graphics display is an unavoidable point of sequential access. However, in reality, an X server can only accept a relatively small number of connections. In addition, the latency associated with each transmission between a parallel process and the X Window server is relatively high. This talk addresses these issues.

Modes. Design of such a library requires tradeoffs among convenience, efficiency, and scalability. We prefer to allow he user to make such decisions by providing multiple *modes* in which the processes created and controlled by the user interact with the X Window server. They are:

Independent Each process connects directly to the X Window server and my issue its own individual drawing commands.

Shared-Server Some processes connect; all may draw. The MPE Graphics Library will deliver the operations to the server internally.

Multi-Server Some processes connect and are dedicated to drawing. This generalizes what is perhaps the most commonly used way of obtaining graphical output from parallel programs, in which one process is responsible from interacting with the X Window server.

Collective Some processes connect, and drawing operations are collective (that is, the drawing operations have the same usage semantics as the MPI collective routines).

All drawing routines may be used in any of the four modes (with restrictions noted below). By using the same interface for all modes, the programmer can easily modify the tradeoffs for scalability and response time without changing the details of the individual drawing commands.

The independent mode is easily understood. In the Multi-Server mode, the `MPE_Open_graphics` call selects some nodes as servers; these make a connection to the graphics display. An MPI communicator that contains only those server nodes is created. Subsequent graphics calls from non-server processes send messages to some server process; graphics calls on a server process both make Xlib calls and check for incoming graphics requests from non-server nodes. Note that for the Multi-Server mode to work, we need the contexts of MPI, since pending server requests must be kept independent from operations such as `MPI_Recv( ..., tag = MPI_ANY_TAG, source = MPI_ANY_SOURCE, ... )`.

The Shared-Server mode is like the Multi-Server mode, except that the server nodes only act as graphics servers. In Collective mode, drawing operations and input operations are synchronous and collective. As in other MPI collective operations, all processes in the associated communicator must make the call.

Summary of Routines in the Library. Here we give a quick summary listing of the routines currently in the MPE Graphics Library. (The full paper will have the functionality of each spelled out in detail.)

```
MPE_Open_graphics - (collectively) opens an X Windows display
MPE_Draw_point - Draws a point on an X Windows display
MPE_Draw_points - Draws points on an X Windows display
MPE_Draw_line - Draws a line on an X11 display
MPE_Fill_rectangle - Draws a filled rectangle on an X11 display
MPE_Update - Updates an X11 display
MPE_Close_graphics - Closes a X11 graphics device
MPE_Xerror( returnVal, functionName )
MPE_Make_color_array - Makes an array of color indices
MPE_Num_colors - Gets the number of available colors
MPE_Draw_circle - Draws a circle
MPE_Draw_logic - Sets logical operation for laying down new pixels
MPE_Line_thickness - set thickness of lines
MPE_Add_RGB_color( graph, red, green, blue, mapping )
MPE_Get_mouse_press - Waits for mouse button press
MPE_Iget_mouse_press - Checks for mouse button press
MPE_Get_drag_region - get ``rubber-band box'' region (or circle)
```

Applications. The MPE Graphics Library is already in use. We have used it for enhancing the output of parallel programs (See the Monte Carlo integration and Mandelbrot examples in [1]). A further use is in conjunction with the MPI Profiling Interface, where we have provided a mechanism for program animation [2].

References

[1] William Gropp, Ewing Lusk, and Anthony Skjellum. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. MIT Press, 1994.

[2] Ed Karrels and Ewing Lusk. Performance analysis of MPI programs. In Jack Dongarra and Bernard Tourancheau, editors, *Proceedings of the Workshop on Environments and Tools For Parallel Scientific Computing*. SIAM Publications, 1994.