Executing and Debugging MPJ Express Programs using the Eclipse IDE

Amjad Aziz, Rizwan Hanif, and Aamir Shafi
{amjad.aziz, rizwan.hanif, aamir.shafi}@seecs.edu.pk
14th May, 2010

The video version of this tutorial can be seen at http://www.youtube.com/watch?v=ROXFfUbgY98

Purpose of the Document

The main objective of this document is to show how MPJ Express programs—in the multicore mode—can be executed and debugged in the Eclipse IDE.

Steps

1. Create a new Java project:
   a. Start Eclipse and go to File → New →Java Project. A window with the label "Create a Java Project" will appear asking for the project name ("MulticoreDebuggerDemo" in our case). Click on the next button.
   b. A new window appears with the label "Java Setting", which allows adding external JAR files to the Eclipse project. Select the "Libraries" tab and click on the "Add External JARs…" button to add the mpj.jar file to the project. The mpj.jar file exists in the lib subdirectory of the MPJ Express software. Click the "Finish" button towards the bottom of the window. The new Java project "MulticoreDebuggerDemo" has been successfully created.

2. Write the source code:
   a. Go to File → New → Class and a window appears with the label "Java Class". Write the class name "HelloEclipseWorld" in the "Name" textbox and press the "Finish" button. The HelloEclipseWorld.java class has been created.
   b. As a demo, write the following source code:

```
import mpi.*;

public class HelloEclipseWorld {
        public static void main(String[] args) throws Exception {
                MPI.Init(args) ;

                int rank = MPI.COMM_WORLD.Rank();
                int size = MPI.COMM_WORLD.Size();

                System.out.println("I am process <"+rank+"> of total <"+
                                                size+"> processes.");
                MPI.Finalize();
        }
}
```

3. Execute the MPJ Express parallel program in the multicore mode:

a. Goto the “Run” menu and select the “Run Configurations…” menu item. A window with the label “Create, manage, and run configurations” appears.
b. Double click the “Java application” launch configuration, which is towards the left side of the window. After this a new launch configuration by the name of “HelloEclipseWorld” has been created that appears on the right side of the windows. This new window has several tabs including “Main”, “Arguments”, “JRE” etc. Select the “Arguments” tab and in the appeared window select the Arguments tab.
c. To execute the MPJ Express HelloEclipseWorld program, we need to specify “-jar ${MPJ_HOME}/lib/starter.jar“ arguments to the VM. But before specifying these arguments in the VM arguments textbox, first set the “MPJ_HOME” environment variable if it does not exist already.
d. To specify the “MPJ_HOME” variable, click on the “Variables…” button below the VM arguments textbox. A window appears with the title “Select Variable”. If the MPJ_HOME variable exists, select it. Otherwise, click on the “Edit Variables…” button that triggers a new window. Click the “New” button on this window and set the value of “MPJ_HOME” variable to the root directory (for example D:\mpj-v0_36) of the MPJ Express software. Click the “OK” button when done.
e. A window with the label “Create, manage, and run configurations” should be visible now. Click the “Run” button to execute the parallel MPJ Express program. Output of the program should appear in the console that typically appears near the bottom of the IDE. A user may change the total number of MPJ Express processes by using the switch “-np” in the VM arguments textbox. For example, to run the same program on four processing cores, add the “-np 4” to VM arguments.

4. Debug the MPJ Express parallel program in the multicore mode:
   a. Goto the “Run” menu and select the “Run Configurations…” menu item. A window with the label “Create, manage, and run configurations” appears. Select the “HelloEclipseWorld” launch configuration.
   b. Click the “Arguments” tab and add the string “-agentlib:jdwp=transport=dt_socket,server=y,suspend=y,address=8000” to the VM arguments textbox. The Eclipse debugger would attach itself with the port specified in this string—the value is 8000 by default and can be changed. Click on “Apply” button and press the “Close” button.
   c. Open the “Debug” perspective of the Eclipse IDE and introduce breakpoints in the HelloEclipseWorld program.
   d. Goto the “Run” menu and select the “Run Configurations…” menu item. A window with the label “Create, manage, and run configurations” appears. Click the “Run” button to execute the parallel program. The following output will appear on the console:

```
MPJ Express (0.36) is started in the multicore configuration
Listening for transport dt_socket at address: 8000.
```

   e. Goto the “Run” menu and select the “Debug Configurations…” menu item. A window with the label “Create, manage, and run configurations” appears. Double click the

"Remote Java Application" launch configuration to create the "HelloEclipseWorld (1)" instance—if it does not exist already. The default port for this instance is 8000 but can be changed if required. Click the "Debug" button.

f. The execution starts but all MPJ Express processes (or threads) hang their execution on the first breakpoint. Now the user can manage the execution of their program by selecting the "Debug" tab in the "Debug" perspective. Under the "HelloEclipseWorld (1) [Remote Java Application]" label, there will be an option corresponding to the main thread. In addition there will be other threads that are equal to the number of processes started by the user by specifying the "-np" switch. Execution of these threads can be controlled by using various debugging options provided by Eclipse IDE.