Introduction to Parallel Virtual Machine (PVM)
Fwu-Shan Shieh
fwushan@eng.auburn.edu

I. About PVM

Parallel Virtual Machine (PVM) is a software system that permits a network of heterogeneous Unix computers to be used as a single large parallel computer. PVM started in the summer 1989 ORNL, and has remained as an on going project funded by DOE, NSF, State of Tenn. It is freeware.

PVM is composed of two parts, the first a daemon called pvmd that resides on all architectures that make up the virtual machine. The daemon runs as a user job on the remote machine, so that if you can log onto a machine, then it can become a processor in the parallel machine. Application programs interface with pvmd through a library of PVM interface routines. This library makes up the second part of the virtual machine.

II. Setting and Running PVM

At Auburn pvm (version 3.2.3) is located in /opt/pvm3. In this directory you find everything there is about PVM (such as examples, PVM source code, user guide, ... etc.). To browse PVM user guide, use the pageview command.

To set up
1. Use user-setup to set up path for PVM
2. Create a directory pvm3 in your home directory
3. Create a directory bin in your pvm3 directory
4. Create a directory SUN4 in your pvm3/bin directory
5. Create a link named lib in your pvm3 directory to /opt/pvm3/lib

   cd ~/pvm3
   ln -s /opt/pvm3/lib lib

To compile program
C: cc -I/opt/pvm3/include -o execname -L../../lib/SUN4 program.c -lpvm3
FORTRAN: f77 -I/opt/pvm3/include -o execname -L../../lib/SUN4 program.c -lfpvm3 -lpvm3
NOTE: Your application should be compiled and placed in your pvm3/bin/SUN4 directory,

To start PVM and application
1. To start PVM, you type pvm [hostfile], pvmd [hostfile], or pvmd [hostfile] &
   The pvm [hostfile] gives you a console that allows for interactive control of the parallel machine. Hostfiles are just lists of machine names so that many machines may be easily started. Each name listed in hostfile is automatically added unless preceded by an "&".
2. To execute pvm application, type program name in any shell (window)
3. To finish PVM
   - type halt for pvm [hostfile]
     The quit command only exits the console and doesn’t kill the daemon. To kill it you must
     restart pvm (it will recognize a daemon is running) and not start another, and then execute
     "halt".
   - type CTRL-C for pvmd [hostfile]

4. Here is the trace of a execution
   lab40{psstfs}48: pvm
   lab30> add lab30
   1 successful
   HOST     DTID
   lab30     c0000
   lab40{psstfs}10: hello
   i’m t40001
   from t80001: hello, world from lab30
   pvm> halt
   libpvm [t40001]: mxfer() EOF on pvmd sock

III. Debugging
    Diagnostic print statements sent to either stderr or stdout from a spawned tasked will not
    appear on the user’s screen. All these prints are routed to a single log file of the form /tmp/pvml.<uid>
    on the host where PVM was started.

IV. A Look at the Message Passing Library & Examples
    Sending a message in PVM
    1. a send buffer must be initialized by a call to pvm_initsend()
    2. the message must be "packed" into send buffer using any number and combination of
       pvm_pk*() routines
    3. the completed message is sent to another process by calling the pvm_send() routine or
       multicast with the pvm_mcast() routine

    Receiving a message in PVM
    1. a message received by calling either a blocking routine pvm_recv() or non-blocking routine
       pvm_nrecv(). There is also a probe function pvm_probe() that returns whether a message
       has arrived, but does not actually receive it.
    2. "unpacking" each of the packed items from the receive buffer
Examples
1. hello.c & hello_other.c (located in /opt/pvm3/examples directory)
2. master1.c & slave1.c
3. spmd.c

V. Getting More Information from World Wide Web (WWW)
   PVM homepage: http://www.epm.ornl.gov/pvm/pvm_home.html