Excursions into Parallel Programming

By Michael Chen

November 1, 2007

TJHSST Computer Systems Lab 2007-2008

Abstract:

With more and more computationally-intense problems appearing through the fields of math, science, and technology, a need for better processing power is needed in computers. The solution can be found through parallel processing, the act of linking together multiple computers and cutting run-time by using all of them efficiently. MPI (Message Passing Interface) is one of the most crucial and effective ways of programming in parallel, despite its difficulty to use at times. However, there has been no effective alternate to this date, and continues as the *de facto* standard of parallel programming. By exploring this rapidly-expanding field, novel approaches can be found to the most complex of problems, even modeling climate change.

Introduction:

-MPI is adaptable and can solve a variety of problems
  -Divide and Conquer, Pipeline, Synchronous
-Parallel programming is a rapidly expanding field
-High computational power
-*De Facto* standard of parallel programming
-Promise for the future
-Current development: DARPA Grand Challenge

Miscellaneous chart. Actual info added later. (screenshot prints out black)

Sample Code and Test Runs:

```
while(1)
{
    double px = 0;
    double py = 0;
    //receives info and sets initial velocity
    MPI_Recv(args,3,MPI_DOUBLE,0,tag,MPI_COMM_WORLD,&status);


    double vx = v0*cos(args[1]*M_PI/180);
    double vy = v0*sin(args[1]*M_PI/180);
    double vw = -args[0];
    //checks to see if parent process ends this process
    if (args[2]==666.0)
        break;
    while (py>=0) //as long as the projectile has height
    {
        //recalculates acceleration and accounts for it
        ax = (vx-vw)*(vx-vw)*rf;
        //acceleration of wind depends on direction of wind
        //with respect to the direction of the projectile
        if (vw<vx)
            ax=-ax;
        //checks if wind velocity is working with/against gravity
        if (vy>0.0)
            ay=-g-rf*vy*vy;
        else
            ay=-g+rf*vy*vy;
        vx+=(ax*dt);
        vy+=(ay*dt);
        px+=vx;
        py+=vy;
    }
    //sets value as the ending location of projectile and sends it
    args[2]=px;
    MPI_Send(args,3,MPI_DOUBLE,0,tag,MPI_COMM_WORLD);
                                                                }
```

Sample code from child process in wind velocity lab.

Procedures and Methodology:

-Uses C, C++, and Fortran
-Flexible, not restricted to certain capabilities
-Many Reference Sources
  -*Parallel Programming with MPI*
  -*Introduction to Parallel Programming*
-Many different levels of code difficulty
-Possible implementation of a graphical interface

Results and Conclusions:

-Future research is necessary
  -blocking vs. non-blocking checkpointing
  -easier debugging and programming
-Large range of modeling problems possible
  -Climate change
  -Molecular biology
  -Artificial intelligence
-MORE WILL BE ADDED