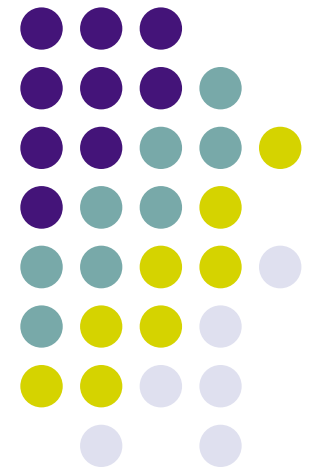
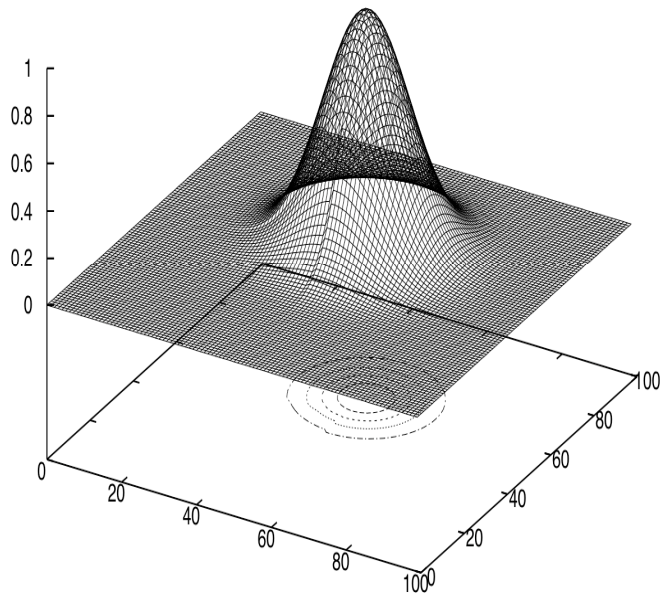
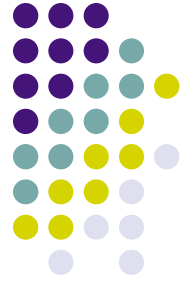


On Parallelizing Advection and Navier- Stokes Simulators

An Introspection

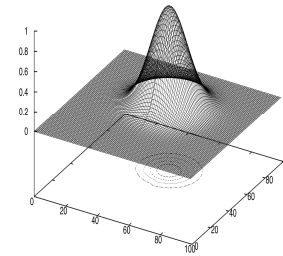


Project Goals



- To parallelize
 - An advection simulator that creates a matrix stored to disk
 - A Navier-Stokes simulator that has the ability to store output to disk

Advection

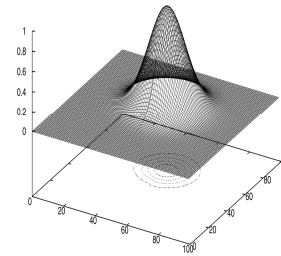


- Challenges
 - Initialization

```
for(i=1;i<nx-1;i++)
  for(j=1;j<ny-1;j++){
    x = (i-1)*dx;
    y = (j-1)*dy;
    gmesh[i][j] = amp*exp(-50*((x-x0)*(x-x0) + (y-y0)*(y-y0)));
  }
```

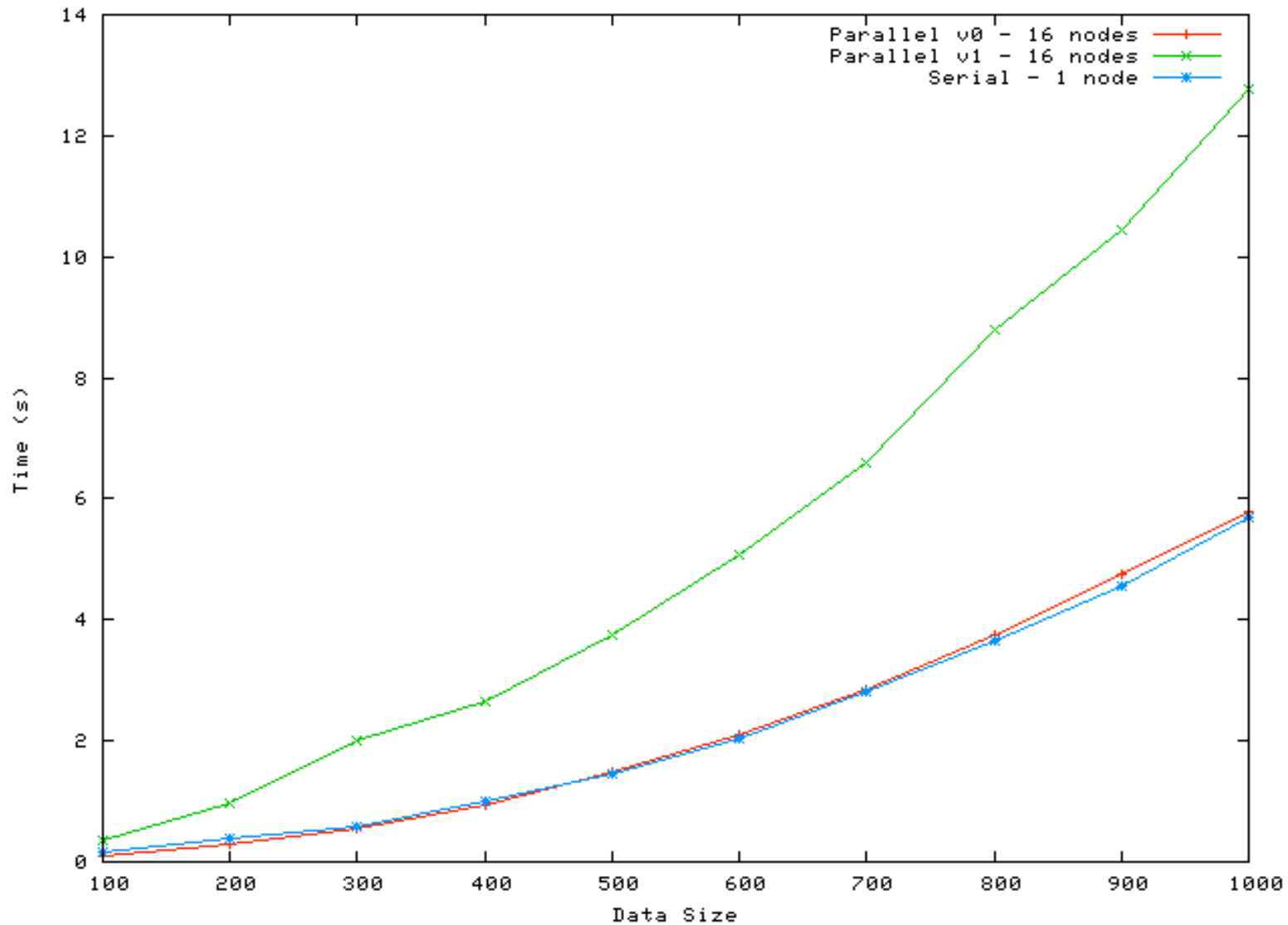
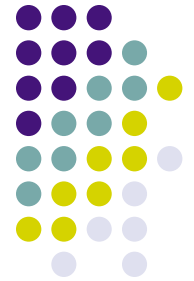
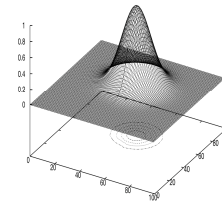
- How to assign values when data is expected to be distributed continuously across the grid?

Advection

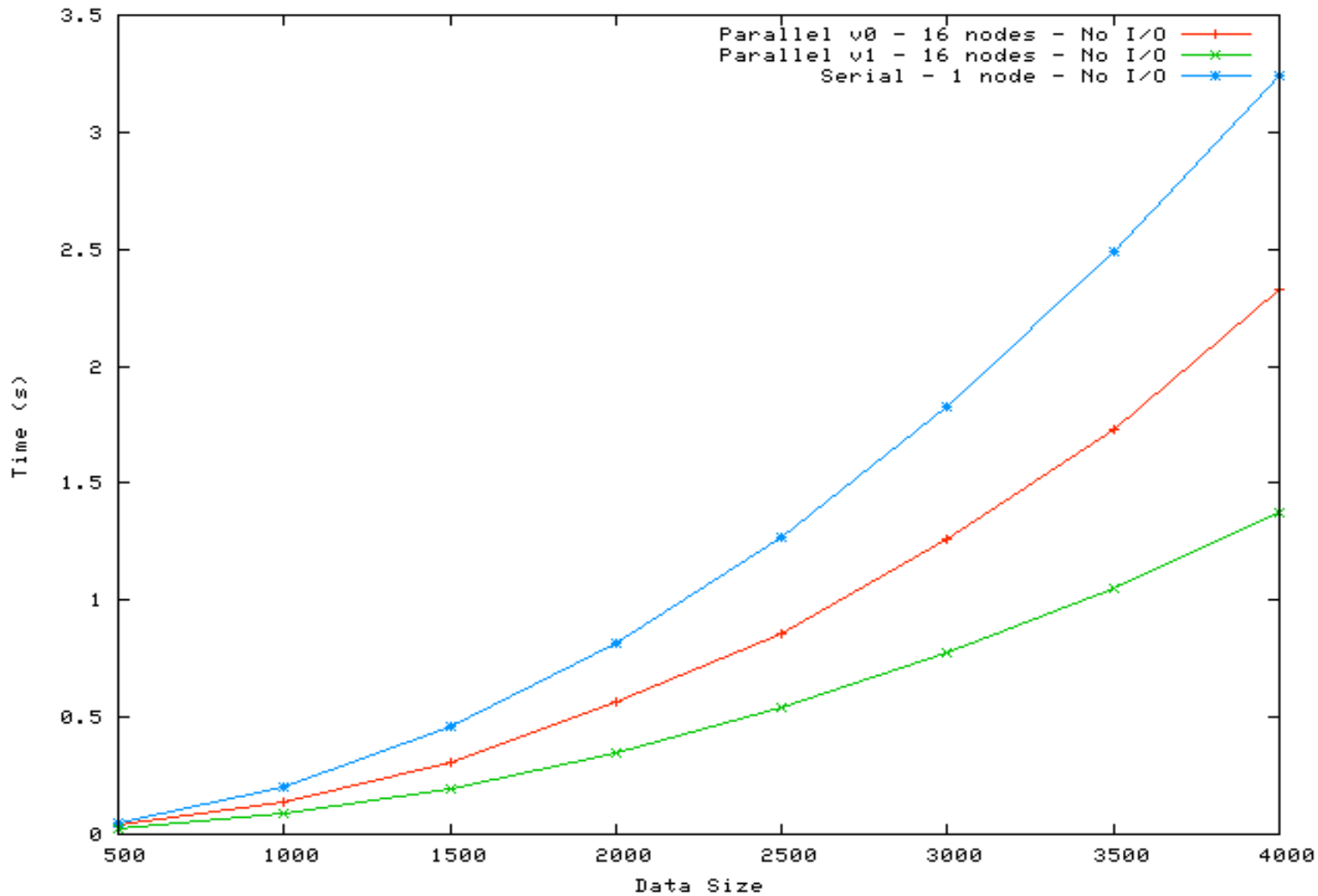
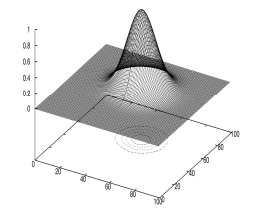


- Challenges
 - I/O - the program needs to output all the data to a file so that it can be plotted...
 - Three mildly unsuccessful attempts:
 1. Send all data to one node, then output
 2. Let each node write to the file in exact place
 - Doesn't work, instead do this sequentially
 3. Use MPI_File routines
 - Bad idea.
 - Lessons learned: mmap

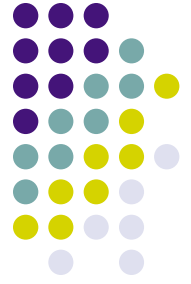
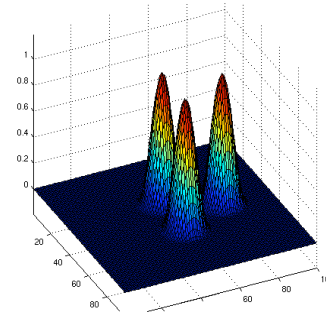
Advection - Results



Advection - Results - No I/O



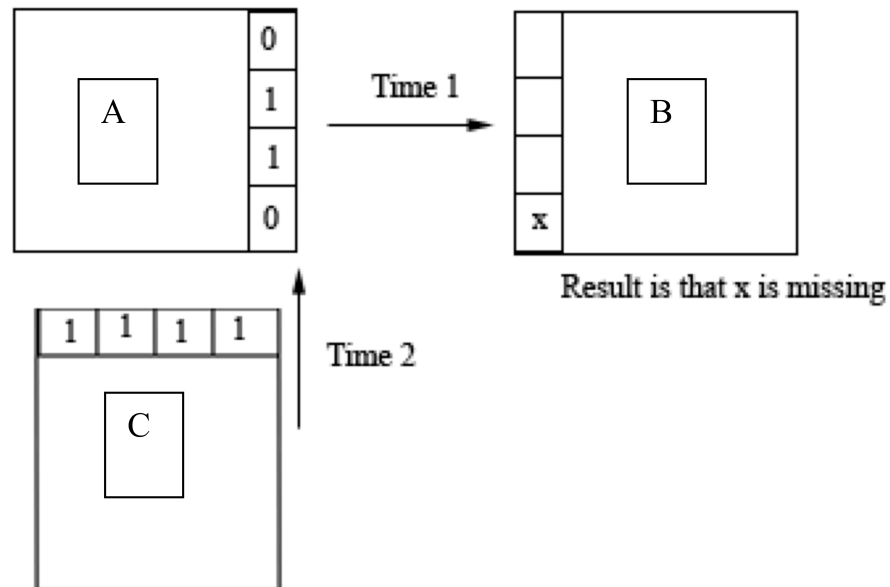
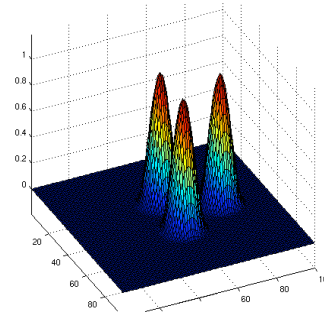
Navier-Stokes



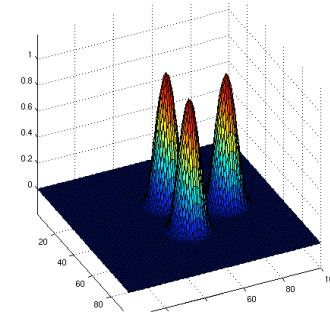
- Challenges
 - Size of program
 - A generous helping of code duplication
 - 20 separate sections that modified data
 - Non-standard indexing and matrices
 - Velocity matrices
 - Performing different work for top, middle, bottom
 - Exchanging ghost corners

Navier-Stokes

- Challenges
 - Ghost corners

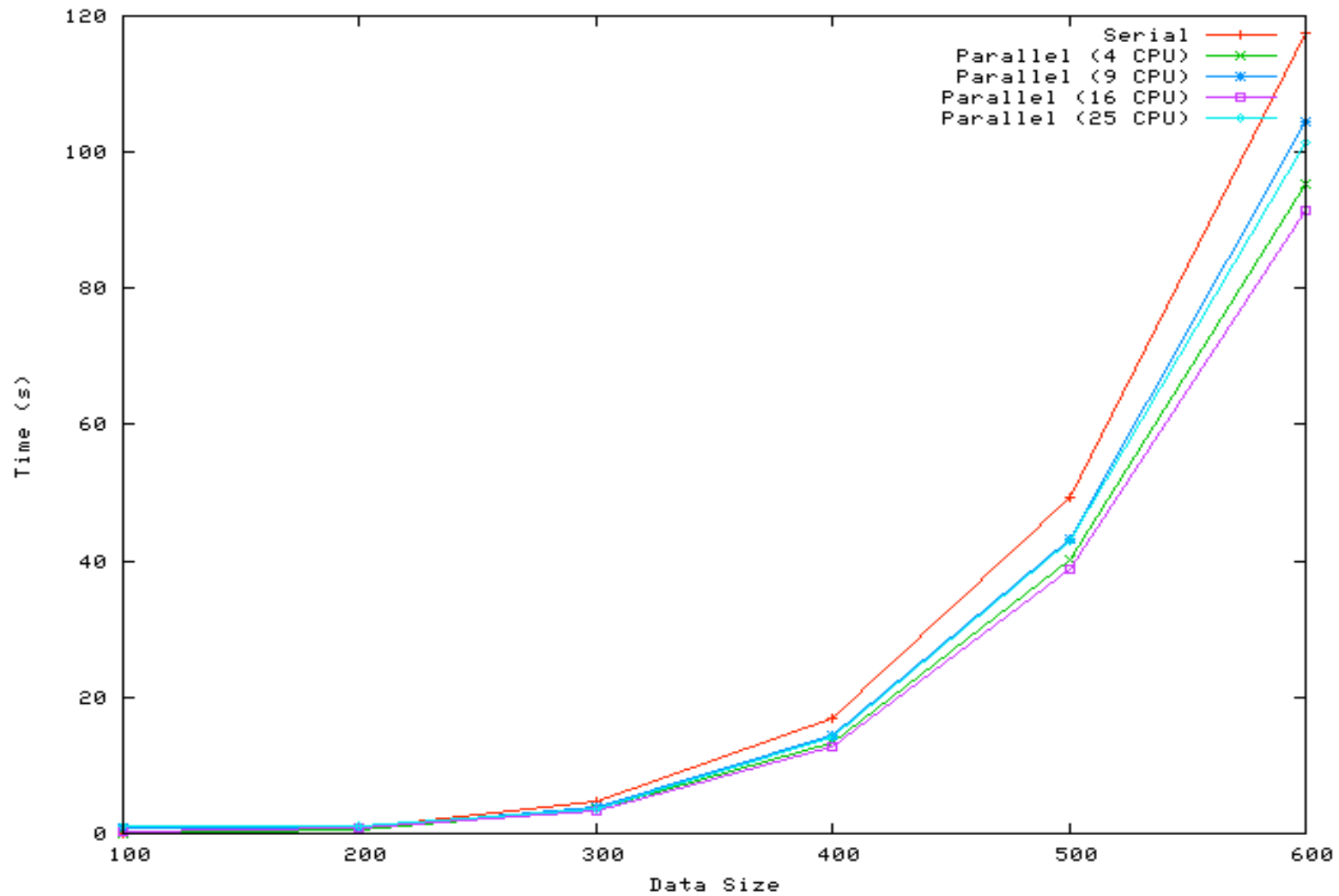
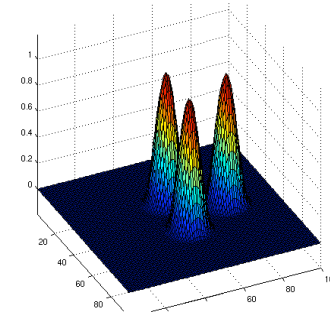


Navier Stokes

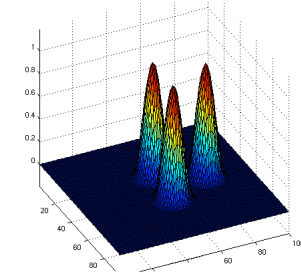


- I/O
 - Used “dd” to create files of appropriate size quickly.
 - Used mmap to memory map files, which allowed each CPU to write data at the same time

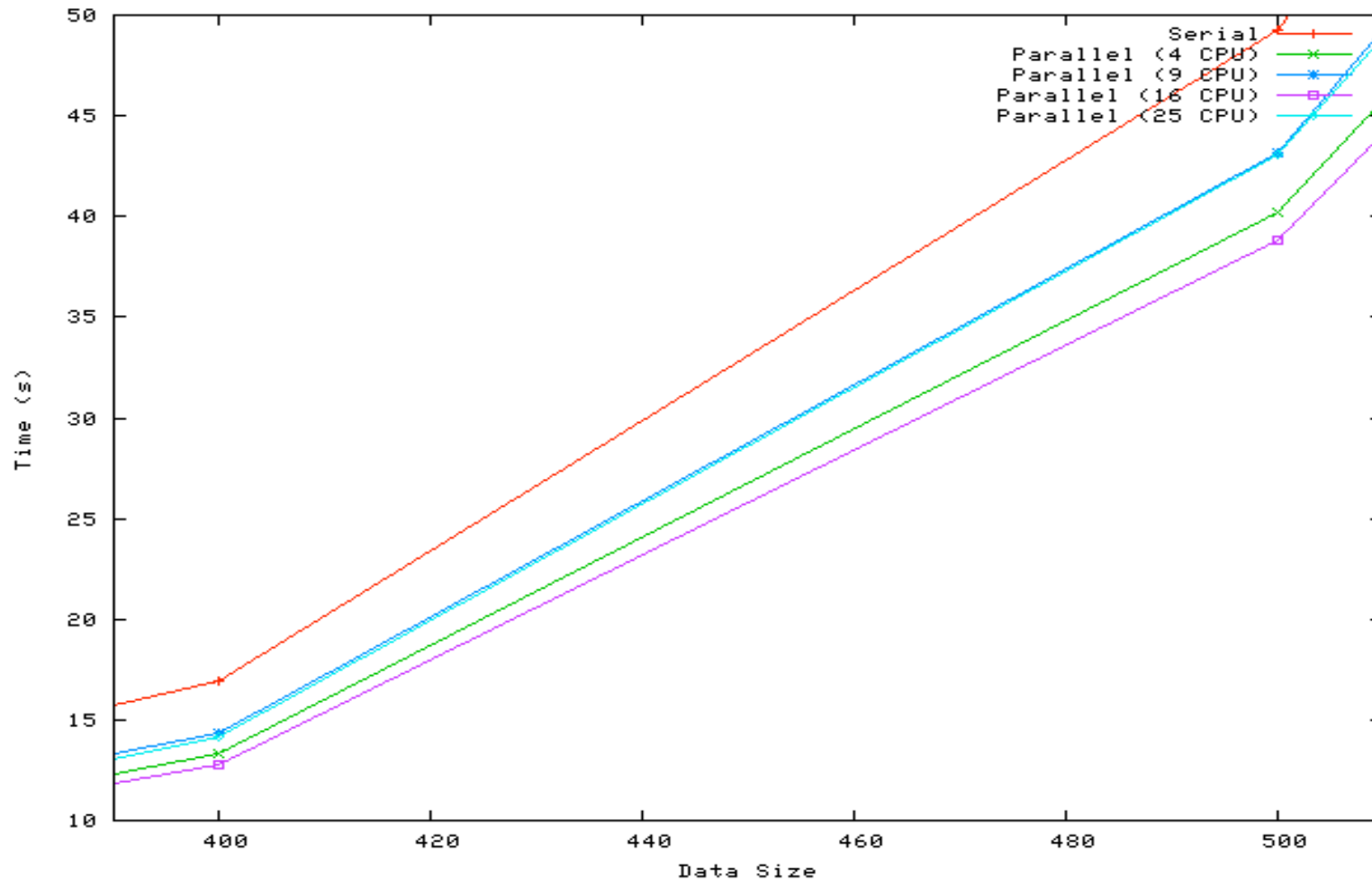
Navier-Stokes Results



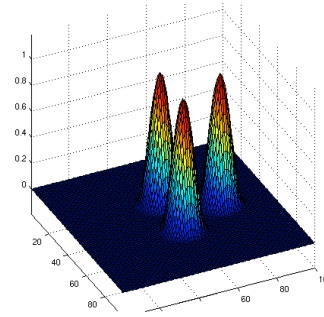
Navier-Stokes Results ?



Can we take a closer look?

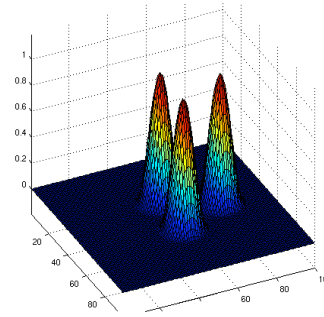


Navier-Stokes Results

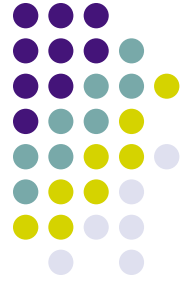


- What happened?
- There's one stage in the program that uses a Poisson solver. The solver is not parallelized (it's in Fortran) and all data had to be collected on one node

Navier-Stokes Results



- How much time is spent in the Poisson solver?
- A test run was done with code profiling
 - Running time: 43.86s
 - Time spent with Poisson solver including gathering and scatter: 42.048s (95.8%)
 - Time spent in Poisson solver only: 40.72s (92.8%)



Conclusion

- Use mmap
- Initialize on separate processors
- Avoid excessive communication
- Avoid parallelizing other people's code =)