

# Comparison of various runs of a simple parallel Jacobi code

Anirudh Modi

April 7, 1998

## 1 Algorithm

The Jacobi method here is applied to solve the Laplace's equation  $\nabla^2\phi = 0$ . Using finite difference, the Laplace's equation can be written as

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} = 0$$

For  $\Delta x = \Delta y$ , we get

$$-4\phi_{i,j} + \phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} = 0$$

So, the Jacobi method simply becomes

$$\phi_{i,j}^{n+1} = \frac{1}{4} [\phi_{i+1,j}^n + \phi_{i-1,j}^n + \phi_{i,j+1}^n + \phi_{i,j-1}^n]$$

where  $n$  is the iteration level.

For the problem we are solving, we have a rectangular domain with constant  $\phi$  (in this case temperature) along two different continuous boundaries ( $\phi = 0$  for one and  $\phi = 100$  for the other which represents the initial temperature) and  $\frac{\partial\phi}{\partial x} = 0$  and  $\frac{\partial\phi}{\partial y} = 0$  in between the two regions.

## 2 Discussion

The code listing and the table of comparisons along with corresponding graphs are attached with this report. The contour plots for the 1, 500 and

1000 iteration cases are also attached. The parallel code speedup for fixed problem size seems to be following *Amdahl's Law* while the speedup for the scaled problem size seems to follow the *Gustafson's Law*. From the calculation of R assuming Amdahl's law holds for the fixed problem size case, we note that the implemented Jacobi code is approximately 90 – 92 percent parallel. The mflops per processor for the scaled problem case is seen to be almost constant at approximately 4.5 mflops. The communication time takes most of the time in the execution of the code (approximately 50 – 60 percent).