CS 359: Parallel Computing Project Proposal

Parallel Implementation of the Conjugate Gradient Method

DATE
30/10/2018

MENTOR
Dr. Surya Prakash

BY
Joshi Niranjan Suhas (160001026)
Naik Prathamesh Umesh (160001037)

Introduction

The conjugate gradient method is one of the most effective algorithms for the numerical solution of large systems of linear equations whose matrix is **symmetric** and **positive-definite**. Since some sparse systems are too large to be handled by a direct implementation, it is usually implemented as an iterative algorithm. Such systems often arise while solving partial differential equations like the Poisson equation. This method is also used in solving various unconstrained optimization problems like energy minimization.

The conjugate gradient method is especially useful when the matrix is sparse because its time and space complexity depends on the number of non-zero elements, rather than the total size.

In this project, we will implement a parallel version of the iterative conjugate gradient method and do a performance analysis.

Methodology

Since this is an iterative method, the result at the next step depends on the result at the current step. Thus, it is not possible to simultaneously compute the intermediate steps.

However, we can parallelize the operations in a step.

Following is the pseudocode for sequential implementation of iterative conjugate gradient method (taken from Wikipedia):

$$
\begin{aligned}
&\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0 \\
&\mathbf{p}_0 := \mathbf{r}_0 \\
&k := 0 \\
&\text{repeat} \\
&\qquad \alpha_k := \frac{\mathbf{r}_k^{\mathsf{T}} \mathbf{r}_k}{\mathbf{p}_k^{\mathsf{T}} \mathbf{A} \mathbf{p}_k} \\
&\qquad \mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k \\
&\qquad \mathbf{r}_{k+1} := \mathbf{r}_k - \alpha_k \mathbf{A} \mathbf{p}_k \\
&\qquad \text{if } r_{k+1} \text{ is sufficiently small, then exit loop} \\
&\qquad \beta_k := \frac{\mathbf{r}_{k+1}^{\mathsf{T}} \mathbf{r}_{k+1}}{\mathbf{r}_k^{\mathsf{T}} \mathbf{r}_k} \\
&\qquad \mathbf{p}_{k+1} := \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k \\
&\qquad k := k + 1 \\
&\text{end repeat} \\
&\text{The result is } \mathbf{x}_{k+1}
\end{aligned}
$$

The above sequential algorithm will be parallelized by using:

1. Parallel sparse matrix-vector product
2. Parallel dot product of Vectors
3. Parallel L-2 norm
4. Parallel copying of vectors
5. Parallel addition of vectors
6. Parallel computation of vector times a constant

System Requirements and Technological Stack

Multicore System.
GCC Compiler (4.6.2 or above)
Openmp along with other stdc++ libraries.

Data Used and Performance Analysis

CSR format for storing sparse matrix. This saves a lot of space.
The matrix A will be real, symmetric & positive definite.

Performance analysis will include

1. Complexity analysis
2. Speedup for various input sizes
3. Convergence to correct value upto desired accuracy
4. Performance for different number of threads and cores used

References

- Wikipedia. *Conjugate gradient method*. (https://en.wikipedia.org/wiki/Conjugate_gradient_method)
- Caraba, E. . *A Parallel Implementation Of The Conjugate Gradient Method*. (pdfs.semanticscholar.org/eaed/aec18d8931d99abe7fbd5b48e3de6997b7be.pdf)
- Rudi Helfenstein, Jonas Koko, "Parallel preconditioned conjugate gradient algorithm on GPU", *Journal of Computational and Applied Mathematics*, (http://www.sciencedirect.com/science/article/pii/S0377042711002196)
- R. P. Bycul, A. Jordan and M. Cichomski, "A new version of conjugate gradient method parallel implementation", *Proceedings. International Conference on Parallel Computing in Electrical Engineering* (https://ieeexplore.ieee.org/document/1115282)