The Ministry of Education and Science of the Russian Federation

Lobachevsky State University of Nizhni Novgorod

Computing Mathematics and Cybernetics faculty

The competitiveness enhancement program

of the Lobachevsky State University of Nizhni Novgorod

among the world's research and education centers

Strategic initiative

“Achieving leading positions in the field of supercomputer technology

and high-performance computing”

INTRODUCTION TO PARALLEL PROGRAMMING

Lecture 16. Parallel Methods for Solving Partial Differential Equations

Nizhni Novgorod

2014

Lecture_16_. Parallel Methods for Solving Partial Differential Equations

Partial differential equations are widely used in various scientific and technical fields. Unfortunately, analytical solution of these equations may only be possible in some special cases. Therefore approximate numerical methods are usually used for solving partial differential equations. The amount of computations to perform here is usually significant. Using high-performance systems is traditional for this sphere of computational mathematics. Numerical solution of differential equations in partial derivatives is a subject of intensive research (see, for instance, Fox, et al. (1988)).

Let us consider the numerical solution of *the Dirichlet problem for the Poisson equation*. It is defined as a problem of finding function $u = u(x, y)$ that satisfies in the domain *D* the following equation:

$$\begin{cases} \dfrac{\delta^2 u}{\delta x^2} + \dfrac{\delta^2 u}{\delta y^2} = f(x, y), & (x, y) \in D, \\ u(x, y) = g(x, y), & (x, y) \in D^0, \end{cases}$$

and takes the value $g(x, y)$ at the boundary D^0 of the domain D when continuous functions f and g are given. Such model can be used for describing steady liquid flow, stationary thermal fields, heat transportation processes with the internal heat sources, elastic plate deformation etc. This problem is often used as a training problem for demonstrating various aspects of parallel computations (see, for instance, Fox, et al. (1988), Pfister (1998)).

For simplicity hereinafter the unit square

$$D = \{(x, y) \in D : 0 \le x, y \le 1\}$$

will be taken as the problem domain.

This lecture deals with the issues of numerical solution of partial differential equations. The most common approach to numerical solution of partial differential equations is the finite difference method; however, it requires large-scale computations. The lecture also describes possible ways to perform parallel computation for network methods in the context of multiprocessor systems with shared and distributed memory.

The description of parallel computation scheme has a special focus on OpenMP, identifies problems related to OpenMP application and the ways to solve them. Interlocking problems can be solved with the help of semaphores while computational ambiguity can be avoided by alternating odd and even rows in the course of processing.

This lecture also deals with the issues of computing on shared memory system. It indicates parallelization principles and describes implementation schemes.

The lecture lists experimental results for each implementation.

Additional information as regards the issues of numerical solution of partial differential equations using finite difference method can be found in [24, 25] and [26].

The issue of memory organization and caching is described in [19].

Task queues and the rules of task selection for synchronization with processor state and extension of the existing task queues are described in [8, 29].

Test questions

1. How is the Dirichlet problem for Poisson equation defined?
2. How is the method of finite differences applied for solving the Dirichlet problem?
3. What approaches determine the parallel computations for the grid methods on shared memory multiprocessor systems?
4. What is the essence of parallel computation synchronization?
5. What are the characteristics of the race condition of the parallel program threads?
6. What is the essence of the deadlock problem?
7. Which method guarantees the grid method determinacy but requires a great additional memory for implementation?
8. How does the computation amount change in case of using the wavefront processing methods?
9. How is chunking applied to reduce computation synchronization?
10. What are the ways to increase the efficiency of wavefront data processing methods?
11. In which way does the job queue provide for computational load balancing?
12. What are the problems to be solved in the course of parallel computations on distributed memory systems?
13. What mechanisms can be involved into the process of data transmission?
14. In which way can the multiple wave calculations be applied for increasing the wave computation efficiency in distributed memory systems?

Practice

1. Develop the first and the second variants of the Gauss-Seidel parallel algorithm. Carry out the computational experiments and compare the execution time.
2. Implement the parallel algorithm based on the wavefront computation scheme. Develop the implementation of the parallel algorithm, in which the block-oriented approach to the wavefront processing method is applied. Carry out the computational experiments and compare the execution time.
3. Develop the implementation of the parallel computation job queue for shared memory systems. It is necessary to provide processing the neighboring blocks on the same processors.

References

1. **Buyya, R.** (Ed.) (1999). High Performance Cluster Computing. Volume1: Architectures and Systems. Volume 2: Programming and Applications. - Prentice Hall PTR, Prentice-Hall Inc.
2. **Chandra, R., Dagum, L., Kohr, D., Maydan, D., McDonald, J., and Melon, R.** (2000). Parallel Programming in OpenMP. Morgan Kaufmann Publishers.
3. **Culler, D., Singh, J.P., Gupta, A.** (1998) Parallel Computer Architecture: A Hardware/Software Approach. - Morgan Kaufmann.
4. **Fox, G.C. et al.**(1988). Solving Problems on Concurrent Processors.- Prentice Hall, Englewood Cliffs, NJ.
5. **Group, W., Lusk, E., Skjellum, A.** (1994). Using MPI. Portable Parallel Programming with the Message-Passing Interface. –MIT Press.
6. **Pfister, G. P.** (1995). In Search of Clusters. - Prentice Hall PTR, Upper Saddle River, NJ (2nd edn., 1998).
7. **Roosta, S.H.** (2000). Parallel Processing and Parallel Algorithms: Theory and Computation. Springer-Verlag,NY.
8. **Tanenbaum, A.** (2001). Modern Operating System. 2nd edn. – Prentice Hall
9. **Xu, Z., Hwang, K.** (1998). Scalable Parallel Computing Technology, Architecture, Programming. – Boston: McGraw-Hill.