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

Lectures 17,18. Parallel Computations for Systems with Shared and Distributed Memory.

Nizhni Novgorod
2014

Lecture_17,18_. Parallel Computations for Systems with Shared and Distributed Memory.

Citius, Altius, Fortius [1], the Olympic motto, is especially true for IT engineering. Implementing an empirical law first formulated in 1965 by Gordon Moore and still holding true is a matter of honour for any hardware manufacturer. The user's/consumer's point of view is best reflected by the following wording: computing system performance doubles every 18 months. We avoided the term "CPU" on purpose, as the end user is not interested in what improves the performance, be it CPU, accelerator or video card: the only thing that is important is better performance for the same money.

However, in the last few years it became clear that computer performance could no longer be improved by increasing CPU frequency, so the manufacturers, having opted for multicoreness as the main development path, had to ask software engineers for help. The existing sequential programs able to use only one core will not run faster on new CPU generation "for free", so parallel programming has to become ubiquitous.

In addition to the above, another Moore's law wording states that the computational performance available to mankind will double each 18 months. A tangible substantiation of this wording is the Top500 list [46] that ranks the world's most powerful computer systems updated twice a year. The 31st edition of Top500 in June 2008 evidenced passing the petaflop/s milestone by IBM Roadrunner [47] whose LINPACK benchmark performance reached 1,026 petaflop/s (the previous teraflop/s milestone was passed by Intel ASCI Red [48] in 1997, so we can see that within 11 years the performance peak grew three decades). The total performance of the systems constituting the 31st edition of Top500 reached 11.7 petaflop/s. Is this much or not? If the real performance of a good PC based on a quad-code CPU is about 20 Gflop/s, the total of Top500 will correspond to 500,000 such PCs. It is clear that this is only the top of the iceberg. According to Gartner, the total number of computers used worldwide exceeded 1 billion in 2008.

The Top500 data help identify typical trends of HPC development. The first version of Top500 was released in June 1993 and contained 249 shared memory multiprocessor systems and 97 single-processor supercomputers; more than 40% of solutions in this list were based on a platform developed by Cray. Only 4 years later, all single-processor supercomputers disappeared from Top500 giving way to the first cluster system with the performance of only 10 Gflop/s (which is 100 times less than that of ASCI Red topping the list); the then-new cluster systems now account for 80% of Top500 and are actually the main way to build supercomputers.

---

[1] Latin for "Faster, Higher, Stronger"

The main advantage of clusters that predetermined their omnipresence, has been the use of mass market hardware and software. Currently, 75% of the systems in the list are based on Intel processors, slightly more that 13% have IBM processors and 11% are AMD-based (each of the two remaining manufacturers, NEC and Cray, account for one system); 81% of the systems use only two types of data communication network systems, i. e. Gigabit Ethernet or Infiniband; 85% of them are Linux-based. As we can see, the list shows little diversity, which is an obvious advantage from mass users' point of view.

However, users would be even more glad to own a desktop or, if worse comes to worst, a deskside supercomputer. So the clusters that brought the DIY supercomputer trend to the HPC industry meet this need in the best possible way. Now it is difficult to say what system can be called the world's first personal cluster. In any case, RenderCube [49] presented a namesake mini-cluster of 4 two-processor system in a compact 42-cm cubicle as early as in 2001.

The HPC personalization trend is developing more and more rapidly and has recently been welcomed by manufacturers of video cards whose performance improved so much that one may want to use them not only for graphical calculation, but as general purpose accelerators, too. The current solutions in this field are provided by NVIDIA (NVIDIA® Tesla™) and AMD (ATI FireStream™) and, being internal devices, demonstrate a stunning (compared to universal processors)peak performance exceeding 1 teraflop/s.

The lecture characterizes the ways to organize parallel computation in a generalized manner and indicates the difference among multitask, parallel and distributed mode of program execution. To illustrate possible approaches, the lectures gives examples of several parallel systems. These examples show that there is a great number of ways to build parallel systems. The lecture also describes computers based on multicore processors to enable high performance computations.

The variety of computer installation requires classification. This lecture describes Flynn's taxonomy, one of the best known classifications based upon instruction and data streams. This is a simple user-friendly classification, however, it will classify almost all multiprocessor computers as *MIMD* systems. For further classification of possible system types, the lecture describes a widely used way to structuralize the multiprocessor computer class thus identifying two important groups of systems with shared and distributed memory, i. e. *multiprocessors* and *multicomputers*. The best known examples of the first group include *parallel vector processors or PVP* and *symmetric multiprocessors or SMP*. Multicomputers include *massively parallel processors* or *MPPs* and *clusters*.

The lecture also pays attention to describing data communication network parameters for multiprocessor computers. It lists examples of network topologies, notes specific features of data data communication networks in clusters and discusses topology parameters that have a material influence on communication complexity of the parallel computation methods.

In its final part, the lecture contains a general description of system platforms for building clusters and shared memory systems.

See additional information on parallel computing system architecture in Hockney and Jesshope (1988), Patterson and Hennessy (1996), Culler, Singh and Gupta (1998), Korneyev (1999), V. Voyevodin and Vl. Voyevodin (2002), Tannenbaum (2002); Xu and Hwang (1998) and Buyya (1999) also give some useful information.

For a review of possible data communication network topologies in multiprocessor systems and techologies to implement them, one may consult Dally and Towles, B.P. (2003).

The issues of construction and use of cluster computing system are described in detail by Xu and Hwang (1998) and Buyya (1999). Practical recommendations on building clusters for various platforms may be found in Sterling (2001, 2002).

Test questions

1. What are the main ways to ensure parallelism?
2. In what way can parallel computer systems differ?
3. What is the basis of Flynn's taxonomy?
4. What is the principle of subdviding multiprocessor systems into multiprocessors and multicomputers?
5. What system classes exist for multiprocessors?
6. What are advantages and disadvantages of symmetric multiprocessors?
7. What system classes exist for multicomputers?
8. What are advantages and disadvantages of cluster systems?
9. What data communication network topologies are most widely used to construct multiprocessor systems?
10. What are the specific features of data communication networks for clusters?
11. What are the main characteristics of the data communication networks?
12. What system platforms may be used for the purposes of building clusters?

Practice

1. Give additional examples of parallel computing systems.
2. Review additional ways to classify computer systems.
3. Review the ways to ensure cache coherency for shared memory systems.
4. Prepare a review of software libraries enabling data transfer operations for shared memory systems.
5. Review a data communication network topology represented as a binary tree.
6. Identify efficiently implementable classes of tasks for each type of data communication network topologies.

Reference

1. **Barker, M.** (Ed.) (2000). Cluster Computing Whitepaper at http://www.dcs.port.ac.uk/~mab/tfcc/WhitePaper/.
2. **Buyya, R.** (Ed.) (1999). High Performance Cluster Computing. Volume1: Architectures and Systems. Volume 2: Programming and Applications. - Prentice Hall PTR, Prentice-Hall Inc.
3. **Culler, D., Singh, J.P., Gupta, A.** (1998) Parallel Computer Architecture: A Hardware/Software Approach. - Morgan Kaufmann.
4. **Dally, W.J., Towles, B.P.** (2003). Principles and Practices of Interconnection Networks. - Morgan Kaufmann.
5. **Flynn, M.J.** (1966) Very high-speed computing systems. Proceedings of the IEEE 54(12): P. 1901-1909.
6. **Hockney, R. W., Jesshope, C.R.** (1988). Parallel Computers 2. Architecture, Programming and Algorithms. - Adam Hilger, Bristol and Philadelphia.
7. Kumar V., Grama A., Gupta A., Karypis G. (1994). Introduction to Parallel Computing. - The Benjamin/Cummings Publishing Company, Inc. (2nd edn., 2003)
8. **Kung, H.T.** (1982). Why Systolic Architecture? Computer 15 № 1. P. 37-46.
9. **Patterson, D.A., Hennessy J.L.** (1996). Computer Architecture: A Quantitative Approach. 2d ed. - San Francisco: Morgan Kaufmann.
10. **Pfister, G. P.** (1995). In Search of Clusters. - Prentice Hall PTR, Upper Saddle River, NJ (2nd edn., 1998).
11. **Sterling, T.** (ed.) (2001). Beowulf Cluster Computing with Windows. - Cambridge, MA: The MIT Press.

12. **Sterling, T.** (ed.) (2002). Beowulf Cluster Computing with Linux. - Cambridge, MA: The MIT Press.
13. **Tanenbaum, A.** (2001). Modern Operating System. 2nd edn. – Prentice Hall
14. **Xu, Z., Hwang, K.** (1998). Scalable Parallel Computing Technology, Architecture, Programming. – Boston: McGraw-Hill.