УДК 004.032

PROSPECTS AND TENDENCIES OF MULTIPROCESSOR SYSTEMS DEVELOPMENT

Іващенко Валерій Петрович, д.т.н., професор
Швачич Геннадій Григорович, д.т.н., професор
Іващенко Олена Валеріївна, ст. викладач
Національна металургійна академія України
Ivaschenko Valery, Doctor of Technical Science, Prof.
Shvachych Gennady, Doctor of Technical Science, Prof.
Ivaschenko Olena, Senior teacher
National Metallurgical Academy of Ukraine, sgg1@ukr.net
Бусигін Володимир Володимирович, аспірант
Дніпровський державний університет ім. О. Гончара
Busygin Volodymyr, PhD
Oles Honchar Dnipro National University, busygin2009@gmail.com

Abstract. Prospects and main tendencies of development and design of modern multiprocessor systems are considered. It is shown that for today the promising solutions are personal computing clusters, which represent a strategic direction for the computing development.

Keywords: multiprocessor systems, cluster systems, data flow, computing systems.

The Beowulf cluster [1] can be considered indicative in terms of the ways of development of multiprocessor systems. This is due to the fact that the development and design of modern multiprocessor systems is accepted to evaluate, analyzing this information project. There are many definitions of the Beowulf cluster. In particular, consider the proposition of Yazek Radaevsky and Douglas Edline: "Beowulf is a multicomputer architecture that can consist of one server node and one or more client nodes connected via *Ethernet* or some other network." So, we mean a system that is built from prefabricated industrial components where *Linux* can be applied, as well as from standard *Ethernet* adapters and switches. It contains no specific hardware components and is easily reproducible. Beowulf also uses software products such as *Linux*, the *Parallel Virtual Machine* (PVM) programming environment, and the *Message Passing Interface* (MPI). The server node manages the cluster as a whole and acts as a server file for client nodes. At the same time, it is a cluster console and a gateway to the external network. Large Beowulf systems can have more than one server node, as well as specialized nodes, such as consoles or monitoring stations.

Obviously, a new approach to designing high-performance Beowulf systems began when its time came. The rapid development of the computer market has provided system developers with new types of components at an affordable price. The industry began to produce fully assembled subsystems (microprocessors, motherboards, disks, and network interface cards). Mass market competition has led to lower prices for such subsystems and improved reliability. The development of free mathematical software (*public domain*) and, in particular, the *Linux* operating system, compilers and programming tools, as well as the *MPI* and *PVM* parallel programming environment, made the hardware largely software-independent. Such projects have formed a rich experience in working with parallel algorithms. Another aspect that stimulated the development of systems of this class was the urgent need for high-performance computing. It can be said that the combination of conditions such as the availability of hardware, mathematical support, experience and expectations creates an environment where the introduction of Beowulf clusters seems a natural step in the path of information evolution.

In addition to increasing the performance of microprocessors, perhaps even more important for the Beowulf project is the optimal price/performance ratio when using network technology. Price optimization and support for the *Linux* operating system for high-performance networks have helped to create balanced complexes, built entirely of finished parts, which made the basic architecture and software simulations more practical.

The first Beowulf cluster was created for the purpose of computations in the *ESS* project, that is, it was intended for Earth and space science. It is designed for people with parallel programming experience. The emergence of a fully clustered user cluster means that they now have a much more efficient and high-performance computing platform. This approach is briefly expressed by the phrase: "Why buy what you can do yourself?" In this regard, users consider building and using the Beowulf cluster an investment, while learning about the specifics of the vendor-created system, they only perceive as a dependency on this provider.

The history of computing clusters began in 1994. The pioneer in this case is the *NASA* - *Goddard Space Flight Center* (*GSFC*), in particular, created on the basis of *CESDIS*. *GSFC* specialists in the summer of 1994 built the first cluster called "Beowulf", which included 16 computers *486DX4*/*100MHz*/*16Mb RAM* and three network adapters running in parallel, with a data rate of 10 Mbps. This complex was to become the computing resource of the *Eart* and *Space Sciences Project* (*ESS*).

Four years later, in 1998, at the *Los Alamos National Laboratory* (USA), astrophysicist Michael Warren and other scientists in the group of theoretical astrophysics built a supercomputer, which was a Linux cluster based on *Alpha 21164A* processors clocked at *533 MHz*. Initially Avalon consisted of 68 processors, later their number was increased to 140. Each node had *256 MB* of *RAM*, a 3 *GB* hard drive and a *Fast Ethernet* network adapter. The total cost of the Avalon project was $313,000, and the *LINPACK* test performance (47.7 *Gflops*) allowed it to take 114th place in the 12th edition of the Top-500 list alongside the 152-processor *IBM RS*/*6000 SP* system. In 1998, at one of the most prestigious high-performance computing conferences in *Supercomputing '98*, *Avalon* developers submitted a report "*Avalon*: *An Alpha/Linux Cluster Achieves 10 Gflops* for $150k". The development discussed in this paper was awarded first prize in "best value for money" category. Now *Avalon system* is actively used in astrophysical, molecular and other scientific computations.

Today, personal computing clusters are a strategic direction for computing development. The need for such systems is caused not only by fundamental restriction of the maximum possible speed of ordinary serial computers, but also by the almost constant occurrence of computational problems, when capabilities of existing computing facilities are always insufficient. These include, in particular, numerical simulation of the processes of hydrodynamics and metallurgical thermophysics [2 – 5], pattern recognition, multi-parameter optimization problems, climate simulation, genetic engineering, integrated circuit design, environmental pollution analysis, solving a wide range of multidimensional non-stationary problems, and the like.

The organization of computational concurrency, when multiple data processing operations are performed at the same time, is usually due to multiprocessing. In this case, you can speed up the process of solving the computational problem and increase the productivity of computations by splitting applied algorithm into information-independent parts and arranging execution of each part of computation on different processors. This approach allows performing necessary computations with less time, and ability to achieve maximum acceleration of this process is limited only by number of available processors and "independent" parts in performed computations.

However, it should be emphasized that the use of concurrency has not yet become widespread, as many researchers have suggested. One of the possible reasons for this situation until recently was the high cost of high-performance systems (only large companies and organizations could afford to buy supercomputers). The current tendency of constructing parallel computing complexes of typical structural elements (microprocessors, memory circuits, communication devices), mass production of which has been mastered by industry, has reduced the influence of this factor, and now almost every consumer can have enough BOS products.

Another and, perhaps now, the main reason for restraining the mass spread of concurrency may be that concurrent computations require a "parallel" generalization of traditional sequential computer-based solution technology. Of course, numerical methods for multiprocessor systems must be designed in the form of parallel and interoperable processes that can be performed on independent processors. Algorithmic languages and system software used in this process should facilitate the creation of parallel programs; organize their synchronization, preventing asynchronous processes and the like.

In view of the above, it can be concluded that parallel computing is an important, promising and attractive area of computing application. In addition, parallel computing is a complex scientific and technical problem. Thus, knowledge of current trends in development of computers and hardware to achieve concurrency, ability to develop simulations, methods and programs for parallel solution of problems in data processing should be attributed to the important qualifications of modern specialist in applied mathematics, computer science and computer engineering.

It should be emphasized that the problem of parallel computing is an extremely broad field of theoretical research and practical work. These problems are divided into the following areas of activity:

– *development of parallel computing systems,* dedicated to their construction principles;

– *analysis of the efficiency of parallel computing*, designed to evaluate the acceleration achieved computing and the degree of use of all the capabilities of computer hardware in parallel ways of solving problems;

– *creation and development of parallel algorithms* for solving application problems in different fields of practical application;

– *development of parallel software systems* related to their mathematical simulation;

– *creation and development of system software* for parallel computing systems based on their mobility (transferability to different computing systems).

The scope of cluster systems today is somewhat narrower than supercomputers of other architecture, but it is no less successfully used to model tasks that characterize a variety of processes and phenomena. It is the development of cluster technology that has made computing performance widely available, which has allowed any institution to take advantage of it.

Cluster solutions are the most economically viable option. Unlike most server systems with shared memory, cluster solutions are easily scalable to higher performance systems. Thus, with increase of computing nodes to provide required computing performance, it is not necessary to buy or design a new system, because old computing nodes can be added to old one and its capacity can be easily increased.

Cluster solutions today have a very good value for money and therefore have a relatively low cost of service. This is achieved through the ability to scale and use standard publicly available components, the price of which is constantly reduced.

In addition, the cluster architecture provides high resiliency of the system: when a single node fails, the cluster does not lose its capacity at all, because new tasks can be run on fewer nodes. The faulty node is easily and quickly removed from the rack and replaced with a new one, which immediately goes into operation. This is possible thanks to the switched topology of modern system networks, where messaging between two nodes can occur in many ways

A cluster is a complex hardware and software complex, so its creation does not end with the unification of a large number of processors in one segment. In order for the cluster to solve tasks quickly and correctly, all the components must be carefully selected in accordance with the software requirements, since the performance of the cluster software depends on the architecture of the cluster itself, the characteristics of the processor, system bus, memory and interconnect.

The approach under consideration distinguishes between two important types of multiprocessor systems - *multiprocessors* or shared memory systems - and *multicomputers* or shared memory systems.

Multiprocessor evaluation takes into account how to build shared memory. A possible approach is to use a single (centralized) shared memory. This provides *uniform memory access* (*UMA*) and is the basis for design of *vector supercomputers (parallel vector processor* (*PVP*) and *symmetric multiprocessors or SMP*). Examples of the first group include the *CRAY T90 supercomputer*, the second group include *IBM Server p690*, *Sun Fire E15K*, *HP Superdome*, *SGI Origin 300*, etc.

Sharing data can also be ensured with physically distributed memory (naturally, the duration of access will no longer be the same for all memory elements). This approach is referred to as non-uniform access to *NUMA (non-uniform memory access*). Among the systems with this type of memory allocate:

– systems where only local cache memory of existing *COMA* (*cache-only memory architecture*) processors is used to display data; e.g. such systems as *KSR-1* & *DDM*;

– systems that provide uniqueness (coherence) to the local cache memory of different processors (*cache-coherent NUMA or CC-NUMA*), include *SGI Origin2000, Sun HPC 10000, IBM / Sequent NUMA-Q 2000;*

– systems that share local memory of different processors without hardware cache coherence support (*non-cache coherent NUMA or NCC-NUMA*), like *CRAY T3E.*

Multicomputers (shared memory systems) can no longer share all available memory (*NORMA*) systems. That is why the proposed approach is useful in making of two important types of multiprocessor computing systems: *massively parallel* (*MPP*) and clusters (clusters). Representatives of first type include *IBM RS/6000 SP2, Intel PARAGON/ASCI Red, Parsytec computer systems*, etc., and examples of clusters include, in particular, AC3 Velocity and *NCSA/NT Supercluster systems*.

It is worth noting the extremely rapid development of cluster-type multiprocessor computing systems. They are a set of separate processors integrated into a network, where the use of special hardware and software allows for unified management (*single system image*), reliable functioning (*availability*) and efficient use (*performance*). However, for parallel solution of computational problems in algorithms it is enough to allocate only large independent parts of computations (*coarse granularity*), which, in turn, reduces the complexity of constructing parallel computing methods and reduces the data flows exchanged by the cluster nodes. At the same time, it should be emphasized that the organization of interaction of computing nodes of the cluster by means of message transmission, as a rule, causes considerable time delays in calculations, which imposes additional restrictions on the types of developed parallel algorithms and programs.

References

1. Официальная страница проекта Beowulf [Электронный ресурс]. – Режим доступа: http://www.beowulf.org/. – Загл. с экрана.

2. Shvachych G.G., Ivaschenko O. V., Busygin V. V., Fedorov Ye. Ye. Parallel computational algorithms in thermal processes in metallurgy and mining // Naukovyi Visnyk Natsionalnogo Hirnychogo Universytetu. Scientific and technical journal, 2018. – № 4 (166), pp. 129 – 137.

3. Shvachych G., Shlomchak G., Moroz B., Fedorov E., Kozenkov D. Automated control of temperature regimes of alloyed steel products based on multiprocessors computing systems // METALURGIJA, 2019. – № 58, pp. 299 – 302.

4. Shvachych G., Moroz B., Pobocii I, Kozenkov D., Busygin V. Automated Control Parameters Systems of Technological Process Based on Multiprocessor Computing Systems // Advances in Intelligent Systems and Computing, Las Vegas, Nevada, USA, Springer, 2019. – V. 2, 763 p.

5. Shvachych G.G., Pobochii I.A., Ivaschenko E.V., Busygin V.V. Research of the problem of compatibility in the multi-processing compound systems // Science review, Vol 1, № 2(9), pp. 19 – 23, February, Warsaw, 2018.