UDC 621.396 **doi: 10.26906/SUNZ.2021.2.074**

Ihor Ivanisenko

Kharkiv National University of Radio Electronics, Kharkiv, Ukraine

DYNAMIC METHOD OF DISTRIBUTED SYSTEM LOAD BALANCING EVALUATE

Abstract. Subject of study is method of estimating resources of the distributed system like a part of scientific problem related to the load balancing and efficient utilization of resources of the distributed system. The paper presents a method of estimating resources of the distributed system, such as network nodes, the processor, memory, and band-width. The proposed method allows to calculate the loading of each node separately in a distributed system and the entire system. Classes of service flows taken into account in the calculation of these resources loading. The complex value of imbalance of load server entered, which taking into ac-count the weight coefficients for processor, memory, and network bandwidth. These weight coefficients allow to select the importance of each network resource (CPU, memory and bandwidth) compared with each other. Also, this method allows to calculate the imbalance of the system servers. Using the method in load balancing allows to distribute requests by servers such way that deviation of the load servers from the average value was minimal, that allow to provide higher system performance parameters (utilization efficiency) and faster processing flows. **Conclusions.** The work proposed a solution to the actual scientific problem of assessing the load of nodes of a distributed system. The proposed method is based on calculating the processor load, memory load, and channel bandwidth by flows of different service classes. Also introduced a complex value of server load imbalance, taking into account weights for processor, memory and network bandwidth. Accordingly, this method allows you to calculate the imbalance of all servers in the system, the average operating time for various balancing algorithms and the efficiency of using the system resources.

Keywords: load balancing; distributed system; multifractal traffic; resource utilization; self-similar flow; imbalance.

Introduction

Problem formulation in general form. Currently, along with the systematic increase in data transmission rates in telecommunications, the share of interactive traffic, which is extremely sensitive to the parameters of the transportation environment, is increasing. To provide the required amount of resources for the transmission of various types of traffic that impose different requirements on the characteristics of the telecommunication network, various mechanisms for ensuring QoS (Quality of service) are used. One of such mechanisms is load balancing [1-3].

The load balancing system solves the problem of ensuring the quality of service and increasing the performance of distributed systems due to the optimal distribution of tasks between the nodes of the computing system. For the most complete use of all available resources of a distributed system, various methods for assessing the load of nodes and the system as a whole are used.

Assessment of the load of a computational node can be done in several ways. One of the analytical methods, which consists in an approximate estimate of the load of each object based on the data on the received tasks. The advantage of the analytical method is that it can accurately estimate the complexity of the problem. The disadvantage of this method is that it can be rather inaccurate if the model for estimating the speed of task execution is inaccurate.

Another way to collect load data is to measure the load of nodes. Most modern machines are equipped with time counters (accurate to microseconds) that can be used to measure the time taken to complete each task.

The advantage of this method is that it is accurate in most cases. The disadvantages include the following: balancing strategies based on this method take into account the past distribution of loads. If the complexity of tasks changes in an unpredictable way, then the method will be inaccurate.

The most famous studies in the field of balancing, theoretical research and development of the fundamental foundations of load distribution, in the creation of a mathematical apparatus, models and control methods for load distribution in distributed systems, were carried out by such scientists as E.I. Ignatenko [2], V.N. Tarasov [3], F. Wang [4], V. Cardellini [1, 5], S. Keshav [6], Xing-Guo Luo, Xing-Ming Zhang [7], Hisao Kameda, Lie Li [8], and many other scientists working on load balancing problems.

The aim of the paper s to develop a method for calculating the load of nodes and the imbalance of a distributed system, based on the estimation of the load of the processor, memory and channel bandwidth.

The analysis of the previous investigations and publications. The task of load balancing in a distributed info-communication system is, based on a set of tasks involving the calculation and transmission of data, and server systems of different capacity, to find a distribution of tasks on servers that provides approximately the same computational load of each server and minimum transmission costs data. To perform this task, various methods and algorithms of load balancing can be used, which take into account the estimates of the load of the computing node.

The most famous research in the field of load management in distributed systems, balancing, theoretical research and development of fundamental foundations of load distribution, in the creation of mathematical apparatus, models and control methods for load distribution were engaged in such scientists as: E.I. Ignatenko, VN Tarasov, F. Wang, V. Cardellini, Xing-Guo Luo, Hisao Kameda, H. Mehta, P. Kanungo, M. Casalicchio, Y.S. Hong, as well as other researchers working on load balancing problems.

Scientists such as S. Keshav, O. Elzeki, M. Reshad, H. Chen, Y. Hu, Shamsollah Ghanbari, Ratan Mishra, Dhinesh Babu, and many others have developed and improved load balancing algorithms.

Experimental research in recent decades has shown that traffic in many multiservice computer networks has self-similar (fractal) properties. The reason for this effect is the distribution of files on the servers, their size, typical user behavior and is largely due to changes in network resources and network topology.

Self-similar traffic causes significant delays and packet losses, even if the total intensity of all flows is far from the maximum allowable values.

There are a large number of publications devoted to the analysis of fractal properties of traffic. Self-similar properties of information flows are found in local and global networks, in particular in Ethernet traffic, ATM, TCP, IP, VoIP applications. K. Park, W. Willinger, P. Abry, M. Taqqu, I. Norros, Potapov A.A., Tsybakov B.S., Shelukhin A.I.

The presence of self-similarity properties in the information flows transmitted by customers significantly affects the efficiency of distributed systems. This plays a particularly important role in the operation of services that provide the transmission of multimedia traffic and real-time traffic. Thus, the task of developing and analyzing load balancing methods that take into account the self-similarity of traffic and loading of each node and the entire distributed system is relevant.

Load balancing method proposed approach

The problem of balancing computational load arises for several main reasons [1, 5, 8]:

- structure of a distributed application is heterogeneous, different logical processes require different computing power;
- structure of a distributed system is also heterogeneous, i.e. different computing nodes have different performance;
- structure of inter-node interaction is heterogeneous, because communication lines connecting nodes can have different bandwidth characteristics.

Depending on the task, you can use static or dynamic balancing [9]. Static balancing is performed before tasks start. However, preliminary allocation of logical processes to processors (servers) has no effect. This is due to the variability of the computing environment (the node may fail), the busyness of the node with other calculations.

One way or another, the gain from distributing logical processes across servers in order to perform parallel processing becomes ineffective. Dynamic balancing provides for the distribution of the computational load on the nodes during the execution of tasks. The software that implements dynamic balancing determines: load of computational nodes; throughput of communication lines; amount of free memory; the frequency of message exchanges between logical task processes, etc. Based on the collected data on tasks and the computing environment, a decision is made on the distribution of tasks between network nodes.

The goal of load balancing can be formulated as follows: based on a set of tasks, including computation and data transfer, and a network of servers of a certain topology, find such a distribution of tasks among servers that provides approximately equal computational load of servers and minimal data transfer costs.

Usually, a practical and complete solution to the load balancing problem consists of four stages [3, 5, 9, 16]: 1) estimation of the load of computational nodes; 2) initiation of load balancing; 3) making decisions on balancing; 4) distribution of tasks.

Let's describe with more details each stage of balancing.

1. Assessment of the system load.

At this stage, the load on each server is calculated by calculating the average utilization of the processor, memory, network bandwidth of the i-th server. The obtained information about the load is used for the balancing process, firstly, to determine the occurrence of an imbalance, and secondly, to determine a new distribution of tasks by calculating the amount of work required to move tasks. Hence, the quality of load balancing work directly depends on the accuracy and completeness of the information.

2. Initiation.

Performing load balancing too often can cause tasks to slow down. The cost of balancing itself may outweigh the potential benefits of balancing it. Therefore, for the productivity of balancing, it is necessary to somehow determine the moment of its initialization. To do this, you should: determine the moment when the load imbalance occurs; determine the degree of need for balancing by comparing the possible benefits of its implementation and the cost of it.

Load imbalance can be determined synchronously and asynchronously. With synchronous imbalance detection, all processors (servers on the network) interrupt their work at certain times of synchronization and determine the load imbalance by comparing the load of an individual processor with the total average load. With asynchronous imbalance detection, each server keeps a history of its load. In this case, the time of synchronization to determine the degree of imbalance absence.

The amount of imbalance is calculated by a background process running in parallel with the tasks.

3. Making decisions in balancing process.

Most of the strategies for dynamic load balancing can be attributed to a class of centralized or a class of fully distributed. With a centralized strategy, the balancer collects global information about the state of the entire computing system and decides to move tasks for each of the servers. A fully distributed strategy runs a load balancing algorithm on each server, exchanging state information with other servers. Tasks are moved only between neighboring processors.

4. Distribution of tasks.

There are many balancing (task distribution) algorithms. However, each of the algorithms has both advantages and disadvantages [10-16]. The most

commonly used load balancing algorithms are the following: Round Robin Scheduling [6, 16], Max-Min Algorithm [5], Compare and Balance [6] and others. You can choose the most suitable algorithm depending on the tasks at hand.

Let's take a closer look at the load balancing method, which takes into account the multifractal properties of additive traffic, and the calculation of the imbalance of resources of the distributed system, which allows to increase the use of system resources by directing heterogeneous information flows to less loaded resources.

Based on the multi-fractal properties of incoming traffic, a dynamic method of balancing traffic is proposed [15]. Depending on the changes in the parameters of the input stream, various methods of traffic management are used. Here is a step-by-step description of the dynamic load balancing method:

Incoming streams are aggregated and processed in accordance with the queue policy, creating a single flow that has characteristics

$$V = \left\{\lambda, h, \mu_{qs}\right\}.$$

1. In the traffic that enters the balancer, a "window" X of fixed length T is allocated.

2. We find traffic intensity, the selective value of the function of the generalized Hurst index $h(q)$, the Hurst parameter value $H = h(2)$, and the range of values of the generalized Hurst index $\Delta h = h(q_{\min}) - h(q_{\max})$ of the segment of traffic in the selected "window".

3. We collect and analyze statistical information by servers and channels: average rate of use of bandwidth $Net_i^{k_i}$ for the time period T, server status $CPU_i^{n_i}$, RAM_i^r, average CPU load, and average usage rate of i-server's memory over a period of time T.

4. Based on the multi-fractal properties of the traffic and the values of the laboriousness of the queries, we calculate the set of vectors of the required resources

$$\mu_{qs}^{new} = \left(CPU, Net, RAM\right)$$

for each qs-class of traffic:

$$\mu_{qs}^{new} = \begin{cases} \mu_{qs}, & \text{if } H \le 0{,}5; \\ \mu_{qs} + (H - 0.5)\mu_0, & \\ & \text{if } 0.5 < H < 0.9,\ \Delta h \le 0.4; \\ \mu_{qs} + (H - 0.5)(\Delta h - 0.4)\mu_0, & \\ & \text{if } 0.5 < H < 0.9,\ 0.4 < \Delta h < 1; \\ \mu_{qs} + \mu_0, & \text{if } H \ge 0.9 \text{ or } H > 0.5,\ \Delta h \ge 1, \end{cases}$$

where μ_{qs} it is determined in accordance with the class of service and the necessary resources, the value μ_0 is selected by the network administrator, taking into account the network status. To reflect the change in the multi-fractal properties of flows, the vectors of the required resources μ_{qs}^{new} are updated at regular intervals and recalculated according to the formula.

The number of required resources does not change ($\mu_{qs}^{new} = \mu_{qs}$) if the traffic is a regular Poisson stream ($H = 0.5$) or has anti-spam properties ($H < 0.5$). With meaning $0.5 < H < 0.9$ and small spread of data ($\Delta h \le 0.4$) the value μ_{qs} increases in proportion to the value of the Hurst index. When the Hurst index $0.5 < H < 0.9$ and the large data scatter ($0.4 < \Delta h < 1$) value μ_{qs} increases, it is proportional to both characteristics.

The number of required resources with the maximum value $\mu_{qs} + \mu_0$ is obtained with the value $H \ge 0.9$ or with persistent traffic ($H > 0.5$) with a range of values of the generalized Hurst index $\Delta h \ge 1$. After transferring the cost of all routes, the message about the state of the paths is sent between the routers.

5. We calculate the distribution of streams by servers based on the listed streams of servers based on the listed values of labor intensity, intensity of traffic, and the state of the server load and communication channels.

6. Based on the data received, the server load is determined in the next step.

7. Distribute traffic to servers, according to the algorithm of balancing within each class of flow.

8. If all traffic could not be distributed, then we allocate the remaining traffic among the amount of resources available:

$$CPU_i^n(T),\ RAM_i^r(T),\ Net_i^k(T).$$

The revaluation is not taken into account by the algorithm, because it does not make any significant changes.

9. We collect data on server's loading

$$CPU_i^n(T),\ RAM_i^r(T),\ Net_i^k(T)$$

and transfer them to the load balancing system to calculate the new distribution of flows.

10. Shift the forward "window" X of length T to the specified shift value ΔT; we analyze the traffic and calculate the next value of server load.

The developed load balancing method should provide a statically uniform distribution of load on the servers, high performance, throughput, fault-tolerance (automatically detecting node failure and redistributing the flow of data among the remaining) and low response time, amount of service information, number of lost data.

Conclusions and directions of future investigation

In the paper proposed a solution to the actual scientific problem of assessing the load of nodes of a distributed system.

The proposed method is based on calculating the processor, memory and channel bandwidth by streams of different classes service.

Also introduced a complex value of the server load imbalance, which takes into account processor, memory and network bandwidth weights.

Accordingly, this method allows you to calculate the imbalance of all system servers, average operating time for various algorithms balancing and efficient use of system resources.

In future we are going to expand number of distributed system involved nodes and modify this method according its physical characteristics.

REFERENCES

1. L. Kirichenko, I. Ivanisenko, T. Radivilova, Investigation of Self-similar Properties of Additive Data Traffic, CSIT-2015 X-th International Scientific and Technical Conference «Computer science and information technologies», Lviv, UKRAINE, 14 – 17 September, 2015, pp. 169-172
2. O. I. Sheluchin, S. M. Smolskiy, A. V. Osin, Self-Similar Processes in Telecommunications, New York : John Wiley & Sons, 2007, pp. 320.
3. Игнатенко Е.И., Бессараб В.И., Дегтяренко И.В. Адаптивный алгоритм мониторинга загруженности сети кластера в системе балансировки нагрузки. // Наукові праці ДонНТУ. – Вип. 21(183). – 2011. – С. 95-102.
4. Chen H., Wang F., Helian N., Akanmu G. User-priority guided min-min scheduling algo-rithm for load balancing in cloud computing // National Conference on Parallel Computing Tech-nologies (PARCOMPTECH). – Bangalore, 2013. – P. 1-8.
5. Cardellini V. A performance study of distributed architectures for the quality of web ser-vices. // Proceedings of the 34th Conference on System Sciences. – Vol. 10. – 2001. – P.213-217.
6. Keshav S. An Engineering Approach to Computer Networking // Addison-Wesley, Read-ing, MA. – 1997. – P. 215-217.
7. Liu J., Luo X., Zhang X., Zhang F., Li B. Job Scheduling Model for Cloud Computing Based on Multi-Objective Genetic Algorithm // IJCSI International Journal of Computer Science. – V.10(1). - № 3. - 2013. – P.134-139.
8. Kameda H., Li L., Kim C., Zhang Y. Optimal Load Balancing in Distributed Computer Sys-tems. – London: Springer, Verlag London Limited. - 1997. – 238 p.
9. Kaur R., Luthra P. Load Balancing in Cloud Computing // Proc. of Int. Conf. on Recent Trends in Information, Telecommunication and Computing. – Association of Computer Electronics and Electrical Engineers. – 2014. – P. 374-381.
10. Sviridov, A., Kovalenko, A. and Kuchuk, H. (2018), "The pass-through capacity redevelopment method of net critical section based on improvement ON/OFF models of traffic", *Advanced Information Systems*, Vol. 2, No. 2, pp. 139–144, DOI: https://doi.org/10.20998/2522-9052.2018.2.24
11. Kovalenko, A.A. and Kuchuk, G.A. (2018), "The current state and trends of the development of computer systems of objects of critical application", Systems of control, navigation and communication, PNTU, Poltava, No. 1 (47), pp. 110–113, DOI : https://doi.org/10.26906/SUNZ.2018.1.110
12. Donets V., Kuchuk N., Shmatkov S. Development of software of e-learning information system synthesis modeling process. *Сучасні інформаційні системи*. 2018. Т. 2, № 2. С. 117–121. DOI: https://doi.org/10.20998/2522-9052.2018.2.20
13. Zykov, I.S., Kuchuk, N.H. and Shmatkov S.I. (2018), "Synthesis of architecture of the computer transaction management system e-learning", Advanced Information Systems, Vol. 2, No. 3, pp. 60-66, DOI: https://doi.org/10.20998/2522-9052.2018.3.10
14. Ruban, I.V., Martovytskyi, V.O., Kovalenko, A.A. and Lukova-Chuiko, N.V. (2019), "Identification in Informative Systems on the Basis of Users' Behaviour", Proceedings of the International Conference on Advanced Optoelectronics and Lasers, CAOL 2019-September,9019446, pp. 574-577, DOI: https://doi.org/10.1109/CAOL46282.2019.9019446
15. Kovalenko, A. and Kuchuk H. (2018), "Methods for synthesis of informational and technical structures of critical application object's control system", *Advanced Information Systems*, 2018, Vol. 2, No. 1, pp. 22–27, DOI: https://doi.org/10.20998/2522-9052.2018.1.04
16. Roth G. Server load balancing architectures, Part 1: Transport-level load balancing. – 2008. – Режим доступа: http://www.javaworld.com/article/2077921/architecture-scalability/server-load-balancing-architectures--part-1--transport-level-load-balancing.html.

Received (Надійшла) 06.04.2021
Accepted for publication (Прийнята до друку) 26.05.2020

Динамічний метод оцінки завантаження вузлів розподіленої системи

І. М. Іванісенко

Анотація. Предметом дослідження є метод оцінки ресурсів розподіленої системи як частина наукової проблеми, пов'язаної з балансуванням навантаження та ефективним використанням ресурсів розподіленої системи. У статті представлений метод оцінки ресурсів розподіленої системи, таких як мережеві вузли, процесор, пам'ять та ширина смуги. Запропонований метод дозволяє розрахувати навантаження кожного вузла окремо в розподіленій системі та у всій системі взагалі. Класи потоків послуг також враховуються при розрахунку навантаження цих ресурсів. Введено комплексне значення дисбалансу сервера навантаження, яке враховує коефіцієнти ваги для процесора, пам'яті та пропускної здатності мережі. Ці вагові коефіцієнти дозволяють вибрати важливість кожного мережевого ресурсу (процесора, пам'яті та пропускної здатності) порівняно між собою. Також цей метод дозволяє розрахувати дисбаланс системних серверів. Застосування методу в балансуванні навантаження дозволяє розподіляти запити по серверах таким чином, щоб відхилення серверів навантаження від середнього значення було мінімальним, що дозволяє забезпечити більш високі параметри продуктивності системи (ефективність використання) та швидший обробка потоків. **Висновки.** У роботі пропонується вирішення актуальної наукової проблеми оцінки навантаження вузлів розподіленої системи. Запропонований метод заснований на обчисленні навантаження процесора, навантаження пам'яті та пропускної здатності каналу за потоками різних класів обслуговування. Також було введено комплексне значення дисбалансу навантаження сервера з урахуванням ваги процесора, пам'яті та пропускної здатності мережі. Відповідно, цей метод дозволяє розрахувати дисбаланс усіх серверів в системі, середній час роботи різних алгоритмів балансування та ефективність використання системних ресурсів.

Ключові слова: балансування навантаження, розподілена система, мультифрактальний трафік, використання ресурсів, самоподібний потік, дисбаланс.