Load balancing in data distribution systems

O V Denisov and E O Vikulov
Omsk State Technical University, 11, Mira ave., Omsk, 644050, Russia
e-mail: olegdenisov95@yandex.ru, vikuloveo@gmail.com

Abstract. The issue of distribution of large amounts of information on server stations is researched. The analysis of existing methods for distributing data of heavily loaded web applications is carried out. A system for computing load distribution using a balancing server is proposed. A description of the software necessary for the functioning of the system and server management interface is provided.

1. Introduction
The distribution of tasks and data between several network devices (for example, server stations), carried out by the load balancing method [1], makes it possible to rationally use resources and reduce the time for the processing of requests in the information system. In modern client-server applications, signal transmission and reception is not a stable process. When a task hits the server, its execution may be blocked by another task in anticipation of shared resources. Task execution time is also a variable. Data transmission over the network occurs with delays, the value of which depends on both the communication protocol and the congestion of the channels.

Load balancing between server stations can be performed using specially designed software. There are various approaches to the distribution of tasks and data between servers, for example, based on the Round Robin circular distribution algorithm [1], based on the current server load [1, 2], etc. These algorithms help rationally allocate resources in the information system and reduce the processing time of each request. But the above algorithms do not take into account (in whole or in part) the parameters and capabilities of server stations.

The article considers the development of an automated system (AS) for the rational distribution of computing load between servers, taking into account the characteristics and status of the servers. Description and study of the algorithms of the proposed system are presented in previous works of the authors [3]. In this paper, we consider some of the existing approaches to load balancing server systems, and analyse their advantages and disadvantages. The main content of the article is devoted to the description of the server-balancer software that provides load balancing on server stations taking into account the data received from them.

2. The problem statement
Currently, there are various systems for distributing large amounts of data. Widespread data delivery and distribution networks 1, CDN (Content Delivery Network), [3] consists of geographically distributed multifunctional platforms, the interaction of which allows to process data and satisfy user requests for information. Another approach to load balancing in information systems is implemented by using data balancing systems that distribute data using a balancing server, which receives a request for data loading and selects a server suitable for data storage [3].

When performing research on approaches to the distribution of large amounts of data, the existing monitoring and control systems for the distribution of load on server stations are considered. For some of the most widely used systems, such as Icinga [4], Cacti [5], Nagios [6], Table 1 and Table 2 present the results of analysis and comparisons of the main characteristics.
Table 1. Characteristics of existing load balancing systems

| Advantages of a surveillance system | The name of the program |
|-----------------------------------|-------------------------|
|                                   | Icinga | Cacti | Nagios |
| Low CPU utilization               | +      | +     | -      |
| Free software                     | +      | +     | +      |
| Open source                       | +      | +     | +      |
| High speed program                | +      | +     | -      |

Table 2. Feature comparison of existing load sharing solutions

| System functions and parameters   | The name of the program |
|-----------------------------------|-------------------------|
|                                   | Icinga | Cacti | Nagios |
| Work with the database            | +      | +     | +      |
| CPU utilization                   | –      | –     | +      |
| Server Action Notification        | +      | +     | +      |
| Boot hard drive                   | +      | +     | +      |

The analysis and comparison of existing monitoring and control systems for load distribution led to the following conclusions:
- all considered systems allow storing data about servers and have the ability to make decisions about choosing a server for downloading information;
- all systems reviewed have the ability to work with databases. They can process information, receive notifications of actions on servers, and operate with minimal hard disk load;
- Nagios system uses CPU utilization to ensure stable operation during data processing, which determines its advantage over other systems;
- low CPU utilization is typical for Icinga and Cacti systems. But at the same time, the advantage of these systems is the high speed of the program.

It should be noted that all considered systems have the following advantages: open source code and availability for free use. The disadvantages of the systems considered are the need to adapt the code for user tasks and the complexity of forming functions that allow you to choose a server based on complete information about its status and characteristics of end users.

To solve the problem of load balancing, taking into account the states and characteristics of server stations, it is necessary to develop a load balancing system. The developed system should carry out information and control functions, enabling employees working with it to receive information about the status of servers and end users. An automated system that provides a complete list of servers that are available provides technical support for the entire server complex.

It should be noted that the developed system provides the ability to access the required information, i.e. an authorized user (who has a certain set of rights) will be able to view the list of servers and check their status. Also, a timely response to the occurrence of errors on the server will make it possible to restore its operability on time. In addition, the AS will eliminate such a drawback of the considered systems as the complexity of forming functions for selecting a server, taking into account the status of the servers and the characteristics of end users.
Thus, to solve the problem of load distribution, taking into account the status of server stations and their characteristics, a load balancing system was developed, including an operation algorithm and software that implements the functions of the system.

3. Theory. Load balancing algorithm

Consider the algorithm of the load distribution system in the information system that redirects client requests to the least loaded or most suitable server from the group of machines on which copies of the information resource are stored, figure 1.

![Flowchart](image)

**Figure 1.** The load balancing algorithm

In order to select a server station, the balancing server receives a list of all servers and begins to sort through them. The balancing server sends a request to each server (from the available list of servers) to receive data on the location of the server station and the current load. If the server station does not respond, then it is marked as idle and is excluded from the list of servers. The server station may be excluded from the list for the following reasons: it is not possible to establish a connection with it, or a response has been received about the server memory overflow. The implementation of the server station selection function is carried out based on the specifics of the ongoing project and the goals that are planned to be achieved. The main goals for which balancing is used in a highly loaded information system are the following:

− timeliness, that is, a guarantee that system resources are allocated for the processing of each request and in this case situations will be excluded when one request is processed and the remaining requests are in standby mode;
− efficiency, which means that all server stations that process requests are as fully loaded as possible;
− reducing the query execution time, that is, ensuring the shortest possible time between the start of processing the request and its completion;
− reduction of response time, that is, the formation of a response to a user request in the least possible time.
Also, the load balancing algorithm of an information system should have the following properties:
− predictability, that is, the user must understand in what situations and under what loads the algorithm will be effective for solving the tasks;
− the ability to ensure uniform workload of all hardware and software resources;
− scalability, which means that the data distribution system must remain operational while increasing the information load.
To increase the operating speed of AS, the authors developed computational algorithms \[7\], that make it possible to distribute data and tasks between servers, which allows to reduce the workload of individual nodes of the information system, to obtain an increase in the speed of work, and ultimately to increase the efficiency of data processing in the system.

4. Results of experimental studies
When developing the AS, an authentication system using a database was implemented using the ExpressJS framework \[8\]. This software platform allows you to implement restriction of access rights and to delimit users of the system, as well as apply extensions to organize role policies.
When working with speakers, unauthorized users cannot perform any operations and switch from the input page to other menu pages: the user will be automatically redirected to the login form. A user without access rights (without a username and password) can be registered only by the system administrator using the database. Also, only the AS administrator has the ability to directly change the set of actions allowed to the user.
To change the availability of actions performed by the system for different users, the AS administrator must go to the action menu bar with the same name as specified in the address bar of the browser and add an array of users for this action (or action). Then these users will have the right to perform the specified operation. MongoDB is used as a database in a data distribution system. \[9\]. Figure 2 shows the program code of the function that adds the list of characteristics to the speakers.

```javascript
stat.set({
  server: config.get('ip'),
  freemem: freeMemory,
  diskFreeSpace: result.free,
  diskUsage: result.used,
  cpuUsage: cpuUsage,
  upTime: systemUptime,
  createdAt: new Date()
});

stat.save()
.then(function () {
  console.log('Server stats saved')
})
.catch(function (error) {
  //send error
  var message = 'Can not save server params';
  console.log(message);
});
```  

**Figure 2.** Program code of the function to add a list of characteristics

The function of connecting to the database is also performed using the framework; in the configuration file of the created application, a connection to the database is established. An example of software
The implementation of the database connection function is shown in Figure 3. The local configuration data was used as parameters.

```javascript
1 var mongoose = require('mongoose);
2 var config = require('..//config').get('db');
3
4 mongoose.connect(config.localhost);
5
6 mongoose.Promise = require('bluebird');
7
8 mongoose.connection.on('error', function (err) {
9   if (err) {
10     throw err;
11   }
12 });
13
14 mongoose.connection.once('open', function () {
15   console.log("MDB connected");
16 });
17
18 module.exports = mongoose;
```

Figure 3. The function for connecting to the database

Consider some of the main functions and actions implemented by the system. Each action implemented by the AS corresponds to a separate operation (action) of one or another controller. Each controller function performs certain actions, interacts with the model and returns control to the file view. In order to describe the internal operation of the system, we describe the actions of all the controllers used and the models and types of actions they cause. During the development, three main controllers were created that describe all the functions performed. These functions correspond to the models: Server, Statistic, User. Table 3 provides descriptions of each of the created models.

| Action     | View  | Model   | Function              |
|------------|-------|---------|-----------------------|
| Login      | login | Users   | Login                 |
| StatAdd    | add   | Statistic | Adding Statistics    |
| StatRemove | delete| Statistic | Delete statistics     |
| ServerAdd  | add   | Server  | Adding a server       |
| ServerDelete | delete | Server  | Server removal        |
| ServerUpdate | update | Server  | Server update         |

During the development of AS, several forms of display were created that give the user the opportunity to quickly receive the necessary information, edit it and present it in a convenient form. The form of the developed interface is shown in Figure 4.
Figure 4. User interface: list of server stations

Figure 4 shows that the list of server stations is displayed in tabular form. The page contains several basic controls, such as buttons: add, delete, update and view. The table shows the identifiers of the servers in the AS, their names and descriptions. This table structure is designed to build a clear and understandable association of the server, its description and number. In addition to the add server button, there is also an edit and delete button; these buttons are located at the beginning of the table and have text prompts. The statistics page is intended for server monitoring; This interface page is shown in figure 5.

Figure 5. Server statistics page

5. The discussion of the results

As a result of the system, we get a large amount of data on the status of server stations. Using methods for analyzing large amounts of data, new patterns can be obtained in the data on the status of server stations. We believe that it is advisable to use the multidimensional statistical procedure for cluster analysis for the final processing of data. So we can divide the received data into groups with similar
parameters and add each new application to one of the groups, thereby determining a suitable server for it [10].

6. Conclusion
The paper describes the results of the development of an automated load balancing system in highly loaded computing systems. A comparative analysis of the existing data distribution methods of highly loaded web applications is carried out. It is noted that these technologies do not take into account the hardware status of server stations, their workload and speed, which negatively affects the speed of data delivery to the user. A load balancing system in highly loaded systems using a balancing server that implements a choice of servers based on an analysis of data on their dynamic characteristics is proposed. The results of the selection of hardware and software tools necessary for the implementation of the development of AS are presented. As a result of the work, the server-balancer control interface and the software necessary to perform the required AS functions have been created, which allows rationally distributing large amounts of information system information to server stations.

References
[1] Vikulov E O, Denisov O V, Denisova L A 2018 Data distribution system: preparation of server stations data J. Phys.: Conf. Ser. vol 1050 p 012097
[2] Vikulov E, Denisov O, Meshcheryakov V 2018 Event-Driven Simulation of Server Stations Load Balancing International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM) pp 1–4
[3] Wilson S, Prakash P and Deepalakshmi P 2017 Server-based Dynamic Load Balancing IEEE Networks & Advances in Computational Technologies (NetACT) International Conf. (Thiruvanathapuram)
[4] Viranch M. 2013 Icinga Network Monitoring (Packt Publishing)
[5] Josephen D. 2013 Nagios: Building Enterprise-Grade Monitoring Infrastructures for Systems and Networks (Prentice Hall)
[6] Urban T. 2017 Cacti Beginner’s Guide - Second Edition: Leverage Cacti to design a robust network operations center (Packt Publishing)
[7] Vikulov E O, Denisov O V, Meshcheryakov V A, Denisova LA 2020 Simulation of data distribution between server stations using fuzzy technologies J. Phys.: Conf. Ser. vol 1441 p 012175
[8] Mardan A 2014 Pro Express.js: Master Express.js: The Node.js Framework For Your Web Development (Apress)
[9] Bradshow A, Brazil E, Chodorow K 2019 MongoDB: The Definitive Guide: Powerful and Scalable Data Storage (O’Reilly Media)
[10] Vikulov E O, Denisova L.A. 2019 Data distribution system: clustering based on neural network technologies IOP Conf. Ser.: Mater. Sci. Eng. 2019 vol 537 p 052030.