Dynamic routing algorithms and processing methods of data flows for cloud applications and services

Denis Parfenov

PCT 2017
Task of the research

1) structural model of the software-defined infrastructure of the virtual data center
2) model of cloud based service-oriented application
3) resource model of the software-defined infrastructure of a virtual data center
4) model of software-configurable scalable data storage
5) algorithmic solutions for routing data flows of cloud-based service-oriented applications
The structure models of infrastructure of virtual data center

VirtualDC={Cloud,Snova,Snas,Srasp,Sstg...}

Snova={VM1,VM2,...VMk}

Snas={VMimg1,VMimg2,...VMimgz}

VM={App1,App2,...Appn}

VMimg={OS1,OS2,...OSr}

Sstg={RDisk1,RDisk2,...RDiskd}

Srasp={Rtask1,Rtask2,...Rtaskf}

Pic. 1 – Decomposition of main resources of virtual data center

Cloud=(State, Mem, Disk, Diskn, Core, Lan) (1)

Characteristics of cloud systems

where:

State – state of the object in cloud system {“on”,“off”};
Mem \in N – amount of RAM installed for node;
Disk \in N – amount of disk space storage devices installed for node;
Diskn \in N – number of storage devices installed for node;
Core \in N – number of processor cores node;
Lan \in N – maximum bandwidth

VirtualDC(t) = (Node(t), Connections(t)), (2)

where:

Node(t) = \{Node_1, Node_2, ..., Node_n\} active elements of the sets Snova, Sstg, Snas, Srasp, Cloud;

Connections(t) = \{Connections_1, Connections_2, ..., Connections_t\} active users connect to cloud applications.
The base model of resource in data center

\[ R = (R_1, R_2, \ldots, R_n) \]  

(3)

each resource \( R_j, j = 1 \ldots N \) characterized by:

• cost \( S_j \) which is expressed in units carrying capacity and is appointed by the resource based on the current state of the data center infrastructure;
• specify the maximum execution time \( T_{\max j} \);
• access priority / starting order \( p_j \);
• a subset of \( H^R_j = (H^R_{j,1}, H^R_{j,2}, \ldots, H^R_{j,k}) \) which is a set of parameters that describe the resource, in this case \( H^R_{j,i} \) - characterized by the available volume for each of the parameters;
• subset of \( E^R_j = (E^R_{j,1}, E^R_{j,2}, \ldots, E^R_{j,k}) \), which represents the efficiency of resource usage for each of the parameters;
• data vector of allowable execution time of the task \( PV(R_j) \);
• consider the set of nodes of the data center \( WN = (WN_1, WN_2, \ldots, WN_l) \);

\[ O = (O_1, O_2, \ldots, O_m) \]  

(4)

each task \( O_m, m = 1 \ldots M \) characterized by:

a set of connected services \( Y = \{y_{m,j}\} \);

subset \( H^O_m = (H^O_{m,1}, H^O_{m,2}, \ldots, H^O_{m,k}) \) set of parameters required to start the service, as the main of which are the characteristics that affect the quality of service (maximum execution time, response time, etc.).
The model of cloud application

\[ CloudAppl = (G, V), \]

\[ G = (\text{Res}, NAppl, Utime, SchemeTask), \]

where \( \text{Res} \) are the resource requirements;

\( NAppl \) is the number of application instances;

\( Utime \) is the estimation of the users request execution time;

\( SchemeTask \) are communication schemes of data transmission between sources and computing nodes.

\[ V = (u, v, Tdata, Mdata, Fdata, Vdata, Qdata), \]

where \( u \) and \( v \) are linked vertices;

\( Tdata \) is the type of transmitted data;

\( Mdata \) – the access method to the information source (REST, JSON and others);

\( Fdata \) – the physical type of accessed object;

\( Vdata \) – the traffic volume estimated on the accessed data (in Mb),

\( Qdata \) – the requirements for the QoS.
The model of cloud service and cloud resource

\[ \text{CloudServ} = (\text{AgrIP}, \text{NameServ}, \text{FormatIN}, \text{FormatOUT}), \]  

where \text{AgrIP} is the address of aggregation computing node; 
\text{NameServ} – the service name; 
\text{FormatIN} – the required format of input data; 
\text{FormatOUT} – the required format of output data.

\[ \text{Cloud Res} = (T \text{Res}, \text{Param}, \text{State}, \text{Core}, \text{Rmem}, \text{Hmem}, \text{Lib}), \]  

where \text{TRes} is the type of resource; 
\text{Param} – the set of parameters; 
\text{State} – the state of resource; 
\text{Core} – the number of computing cores; 
\text{Rmem} – the size of RAM; 
\text{Hmem} – the size of disk; 
\text{Lib} – the libraries requirements.
The model of software-defined storage

\[ S_{stg, ki} = \left( \text{Max} V_{ki}, P_{stg, ki}, \text{Vol}_{ki}(t), \overline{R}_{ki}(t), \overline{W}_{ki}(t), s_{stg, ki}(t) \right) \]  \hspace{1cm} (10)

Where

- \( \text{Max} V_{ki} \in \mathbb{N} \) the maximum storage capacity in Mb;
- \( P_{stg, ki} = \{ p_{stg, kij} \} \) the set of network ports;
- \( \text{Vol}_{ki}(t) \in \mathbb{N} \cup \{0\} \) the available storage capacity in Mb;
- \( \overline{R}_{ki}(t) \) read speed;
- \( \overline{W}_{ki}(t) \) write speeds;
- \( s_{stg, ki}(t) \in \{ "online", "offline" \} \) state of storage.
The simulation model of infrastructure of virtual data center

**Pic. 1.** Queuing system scheme of the infrastructure of virtual data center
The data flow model in the virtual data center

\[ x_{i,j}(t + \Delta t) = x_{i,j}(t) - \sum_{k=1}^{K} \sum_{l=1}^{N} s_{i,j}(t)u_{i,l}^{j,k}(t) + \sum_{m=1}^{N} s_{m,i}(t)u_{m,l}^{j}(t) + y_{i,j}(t) \]  

(11)

\( N \) – the number of virtual nodes within the network;

\( K \) – the number of application types within the network;

\( s_{i,j}(t) \) – capacity of the channels between \( i \)-th computing node and \( j \)-th storage system \((i \neq j)\);

\( y_{i,j}(t) \) – the traffic volume (the number of user requests) at the moment \( t \) on the virtual node \( i \)-th and intended for transferring to the storage system \( j \)-th;

\( u_{i,l}^{j,k}(t) \) – the part of the channel transmission capacity in a certain segment of the network \((i, l)\) at the moment \( t \) for the user request flow to the application of type \( k \), working with storage \( j \);

\( 0 \leq u_{i,l}^{j}(t) \leq u_{i,l}^{j,(\text{max})} \leq 1; \sum_{l=1}^{N} u_{i,l}^{j}(t) \leq \varepsilon_{i,l}^{j,k} \leq 1 \)  

(12)

\( u_{i,l}^{j,(\text{max})} \) the limit of channel capacity available for the computing node \( i \)-th software-defined network segment \( l \) for traffic transfer to storage \( j \)-th;

\( \varepsilon_{i,j} \) is the part of the channel capacity for the compute node \( i \)-th in software-defined network segment \( l \) for the transmission of user requests to the application of the type \( k \) to storage \( j \)-th;

\( 0 \leq x_{i,j}(t) \leq x_{i,j}^{(\text{max})}; \sum_{l=1}^{N} x_{i,j}(t) \leq x_{i}^{(\text{max})} \)  

(13)

\( x_{i,j}^{(\text{max})} \) maximum allowed queue length on \( i \)-th compute node for processing of incoming traffic to storage system \( j \);

\( x_{i}^{(\text{max})} \) maximum allowed volume of the buffer on compute node \( i \);

\[ \sum_{t=0}^{t-1} \sum_{i=1}^{N} \sum_{j=1}^{N} b_{i,j}(t)u_{i,l}^{j}(t) \rightarrow \text{max} \]  

(14)
The allocation algorithm and dynamic balancing of data flow in the virtual data center

Begin

Getting data

Query type, the level of the target object

Getting information about available resources

Users copies of data and applications

There are resources available?

YES

Priority selection and addition to the queue

NO

NO

There are additional resources?

YES

Creating an additional instance

NO

Message about the lack of resources

NO

Loggin

maintenance request

Edn
The allocation algorithm and dynamic balancing of data flow between storage devices

Begin
Getting data
Segmenting data devices
Recording information in the database
Analysis of the status of devices
Requires redistribution?
YES
Analysis of active users and the formation of the plan
NO
Logging
End
The adaptive algorithm of load balancing and routing data flows

Begin

Analyzing query data and initializing parameters

Building an application placement map

Forming a resource demand map and a route map to network resources

Is the solution found feasible?

Reconfiguration of routes and application of rules in a virtual data center network

Конец
The efficiency of use different storage devices

Pic.2 Result of experiment 1 (read)

Pic.3 Result of experiment 2 (write)

Pic.4 Result of experiment 3 (read and write)
The result of experiment

Pic.5 The result of experiment
Thank you for attention!
Step 1. To split the software-defined network multiple channels in a subset of channels which are included in the tree of routes passing through the ST set segments, a subset of alternative channels passing through the SR set segments.

Step 2. To generate optimal routes in the software-defined network for the data stream of a particular class of applications in a software-defined network cloud platform, on the basis of the weights of communication channels for each of the subsets.

Step 3. For each of the software-defined network channel determine the point of entry into the tree of optimal routes and in a variety of alternative channels.

Step 4. For each network object which is a leaf of the tree, search all the alternative routes with minimum cost, taking into account the channel weights in the previously constructed tree of optimal routes of software-defined network. To bind these lists to a network object, incident reporting channel and located below in the hierarchy.

Step 5. If a network object is not a leaf of the tree, then calculate alternative routes in the software-defined network for this object, taking into account the weight of the channel in the previously constructed tree of the optimum routes and select the best value of the alternative route. This procedure is performed to generate a list of alternative routes in the case of a dynamic load change of the communication channel.

Step 6. For each communication hub form the full list of alternative paths in software-defined network of data center.
Step 7. Analyzing the received protocol information, define if the load changes for some connection has happened or not. If so, move to Step 8, otherwise – to Step 7.

Step 8. Using the list of alternative routes, define if you need to change the route for the current data flow. If so, move to Step 9, otherwise – to Step 13.

Step 9. For a network object with a decreased potential and which alternative routes list include changed metrics connection, define the minimum length route and put the connection, which led to the network object potential decrease into the optimal route tree and the changed path from the optimal paths tree – into the set of alternative SDN routes. When required you should review calculation of channel’s function weight.

Step 10. After the transfer to the alternative software-defined network route, you should define if the potential of other data center network objects, which placed higher in the hierarchy, has decreased. If so, move to Step 11, otherwise – to Step 12.

Step 11. You should define the new minimum length route for every network object which potential has been decreased. If the new minimal length route for every object of the network includes a route from the alternative route list, you need to put this it (route) into the software-defined network optimal routes tree and the route from the optimal routes tree into the set of alternative routes.

Step 12. Design the new optimal route tree in software-defined network of data center.

Step 13. Transfer the current application data flows and services at affordable data center routes, reconsider the tree entry points, and in a variety of alternative routes, reshape the list of alternative routes for each network entity changed. Go to step 7.