A Multi-Constrained Routing Algorithm for Software Defined Network Based on Nonlinear Annealing

Lijie Sheng*, Zhikun Song, and Jianhua Yang
School of Computer Science, Xidian University, Xi’an 710071, China
*Corresponding author, Email: ljsheng@xidian.edu.cn, zksong@stu.xidian.edu.cn, yang0910101027@163.com

Abstract—Multi-constrained Quality of Service (QoS) routing algorithm is always a difficult problem in routing research area, which is a NP problem. Software Defined Network (SDN) is a new network architecture, in which there’s few research on QoS routing. This paper generates the whole network virtual topology according to the characteristics of SDN, and based on the principle of simulated annealing, proposes a nonlinear annealing algorithm, which adapts to the SDN. Firstly we simplify the network topology by Dijkstra-like algorithm, and then introduce the nonlinear energy function, and then iterate the initial solution according to the simulated temperature, until find a feasible path from the source node to the destination node which satisfies the condition. Experimental results show that this algorithm has a higher success rate, better expansibility of network size, and better transplantation for the SDN than traditional QoS routing algorithm.

Index Terms—Quality of Service (QoS), Software Defined Network (SDN), Simulated Annealing, Multi-Constrained Routing

I. INTRODUCTION

Software Defined Network (SDN) [1] is a novel network architecture proposed in recent years. Different from the traditional IP network, SDN separates the network control plane from data plane. The basic idea is to simplify the forwarding function of the hardware and achieve the forwarding decision by software, in the meanwhile, let the hardware focus on packet forwarding by caching these decisions. OpenFlow, as the core technology, [2] separates the control components from the network device and maintains flow table structure on the network devices meanwhile, forwards data packet according to the flow table and manages the generation, maintenance and configuration of flow table by the central controller [3]. OpenFlow technology preliminary realized the design of SDN. SDN technology which bases on OpenFlow, realizes flexible control of network traffic by separating the network equipment control plane from data plane and establishes flexibility controllable Internet through software platform, at the same time, supplies a good platform for the innovation of the core network and provides a new solution for the future Internet technology.

With the high development of the Internet, the Internet has been gradually transferred from a single data transmission network to the integrated transmission network of multimedia information such as data, voice, image and so on. The existing Internet transmission mode is still the single Best-Effort service and hard to meet the different requirements of the multimedia applications and other varied applications for the quality of the network transmission, therefore, the Quality of Service (QoS) is proposed. QoS means to provide various network quality of service according to the characteristics of applications. QoS routing is one of the key technology to guarantee the quality of the network service and a kind of path selection routing mechanism, which can select path according to available resources of network and the QoS requests of business flow. Thus the QoS routing usually needs to meet the multidimensional QoS routing parameters, such as bandwidth, delay, hop count and packet loss rate, etc. In practical applications, QoS routing parameters can often be divided into additive parameters, multiplicative parameters and minimum parameters [4] according to their characteristics. With the rapid development of network technology and the depth researching of QoS routing, algorithms for QoS routing increase fast and all aim to improve the quality of service by finding the optimal path to transmit information, and provide end to end quality of service control and related guarantee [5].

The multi-constrained QoS routing problem is a NP complete problem, which is difficult to find an optimal path or feasible path to meet the conditions. Most of existing algorithms have great limitations, and cannot adapt to the SDN. According to the characteristics of SDN and the advantages and disadvantages of the existing QoS routing algorithms, this paper creatively introduces the simulated annealing and the nonlinear energy function in the QoS routing, and provides an optimized QoS routing algorithm for SDN in order to better transplant to the SDN.

On this occasion, the research about QoS routing in SDN is in great request. We propose a QoS routing algorithm in SDN, which based on the simulated annealing and the characteristics of SDN. By this algorithm, the SDN can also provide the end to end quality of service.
This paper firstly explains the QoS and SDN, and indicates the urgent need of the QoS routing in SDN. Secondly introduces the exist QoS routing algorithm. Thirdly explains the principle of simulate annealing and detailed introduce the algorithm we put forward for the SDN. Fourthly we show the advantages of this algorithm through the simulation. Finally is the conclusion of this study.

II. RELATED WORKS

The multi-constrained QoS routing problem is a NP complete problem. The multi-constrained optimal path problem can be divided into the following categories according to the different constraint conditions:

Constrained Shortest Path Problem (CSP): Find a path that satisfies all the constraints and only one goal can achieve optimal [6].

Multi-Constrained Path Problem (MCP): Find a path that satisfies all the constraints, but no optimization [7].

Multi-Constrained Optimal Path Problem (MCOP): Find the optimal path who can meet M targets and N constraint conditions [8].

The scholars have conducted in-depth researches for the above multi-constrained QoS routing problems and put forward lots of classical algorithms for different types of multi-constrained QoS routing problem.

Because the CSP problem involves only two parameters of delay and cost, so they can be combined into a single parameter and then transformed into a one parameter routing problem. Hilmi Enes Egilmez puts forward LARAC algorithm [9]. This algorithm combined the two parameters of delay and cost by introducing two multipliers. We can gradually approach the optimal path using the iterative method, whose process is very similar to the process of Lagrange relaxation. In the "delay-cost" linear space, the optimal solution can be obtained after many iteration of using Dijkstra algorithm. LARAC algorithm is the most effective method to solve CSP problem at the present stage [10].

Jaffe proposed two algorithms for two-constrained MCP problem [11]. In the second algorithm, Jaffe proposed a linear function to combine the shortest path:

\[ w(u,v) = d_1 w_1(u,v) + d_2 w(u,v). \]  

(1)

In this function \(d_1\) and \(d_2\) are multipliers. Then we use Dijkstra algorithm to obtain the minimum path on the combined parameter calculated by this function. But the shortest path based on linear combination may not satisfy the constraint conditions. Therefore, Jaffe thought about this problem, and proposed a new nonlinear function later:

\[ f(P) = \max \{w_1(P), L_1\} + \max \{w_2(P), L_2\}. \]  

(2)

The nonlinear combination can guarantee that you can find a feasible path if it exists. But the Jaffe algorithm only considered two parameters and the number of constraint parameters are not scalable. And the nonlinear function cannot be solved in linear time.

Lec W C put forward the Fallback algorithm [12] for K-constrained QoS MCP problem. In the calculation, considers only one parameter first, and then, check whether the path meet the other parameter constraints or not after getting the optimal path to this parameter, if meet then stop; otherwise, chooses another one QoS parameter to continue this operation, until you find the feasible path satisfying all QoS constraint conditions, or all QoS parameters have been tried. Fallback algorithm is relatively simple. The computation time is K Dijkstra algorithm in the worst case. But this algorithm does not optimize the path, cannot guarantees the optimality of the path. Even when the feasible path exists, it does not guarantee to find it.

Turgay Korkmaz and Marwan Krunz presented a heuristic H_MCOP algorithm [13], which can provide a good cost function. The cost function [14] considered the entire path:

\[ g(k) = \sum_{k=1}^{K} \left( \frac{w_k(p)}{c_k} \right)^\lambda. \]  

(3)

In the function \(w_k(p)\) is the first \(k\) parameter of path \(p\), and \(c_k\) is the first \(k\) QoS constraint condition. By adjusting the value of \(\lambda\), we can get different computation precision: \(\lambda = 1\) for the linear approximation, \(\lambda = \infty\) for the asymptotic approximation. When \(\lambda\) gradually increases, the calculation precision is higher with larger amount of calculation. The algorithm first calculates the shortest path when \(\lambda = 1\). Then it calculates the shortest path when \(\lambda = \infty\) according to the calculation results in front.

Lianggui Liu and Guangzeng Feng proposed an Multi-constrained QoS routing scheme using Mean Field Annealing (MFA_RS) for wireless mesh networks [15]. Before mean field annealing, it first used a function to combine the wireless link's delay and capacity as the cost function. Then replaced the stochastic process in Simulated Annealing by a set of deterministic equations to iterate initial solution by annealing process, until find the feasible path to meet the constraint conditions. The time this algorithm need to convergence is lesser than the Simulated Annealing because of the deterministic equations and saddle point approximation. But this algorithm is used on the wireless network, because of the big difference of the wireless network and SDN, it’s very difficult to transplant to the SDN.

Baoxian Zhang came up with a bidirectional Multi-Constrained routing algorithm in his paper [16]. This algorithm employed the bidirectional search in the QoS routing in a creative way, which can accelerate the convergence of some QoS routing algorithm and reduce the algorithm execution time.

III. BACKGROUND KNOWLEDGE

A. Definition

Consider a network that is represented by a directed graph \(G(V,E)\), where \(v \in V\) is a node in graph,
$e_{ij} \in E$ is a link in graph. Each link $(i, j)$ has $k$ independent constraint parameters $w_c(i, j)$. This paper only considers the additive parameters: For path $p = v_0 \rightarrow v_1 \rightarrow ... \rightarrow v_n$, satisfy (4) [17].

$$w_c(p) = \sum_{i=1}^{n} w_c(v_{i-1} \rightarrow v_i). \quad (4)$$

where $w_c(p) \in R^k$ is the path constraint parameters, and $1 \leq k \leq K$. For the given source node $s$, destination node $t$, and $k$ QoS constraint conditions: $c = (c_1, c_2, ..., c_k)$, what we need to do is to find the path $p$ from $s$ to $t$, which meets the constraint conditions: $w_c(p) \leq c_k$, where $1 \leq k \leq K$ [18].

The energy function of path $p$:

$$g(p) = \max_{k \in c} (w_c(p)/c_k) \quad (5)$$

where $c = (c_1, c_2, ..., c_k)$ for the $k$ QoS constraint conditions that the feasible path needs to be satisfied. Multiple constraint parameters can be combined into one single constraint parameter by this energy function. The suitable path can be found by using Dijkstra function. But as a nonlinear function, it does not guarantee to find a path with the minimum energy value in polynomial time.

The path from the $s$ to the $t$ is the shortest path to satisfy (6) which goes through the node $u$.

$$w(p) = \sum_{k=1}^{K} w_c(p)/c_k. \quad (6)$$

If $w(p) > K$, the node $u$ will not be in the feasible path from the $s$ to the $t$.

The above principle can simplify network topology, reduce the network size, thus reduce solution space and the algorithm execution time, and make the algorithm adapt to the large and complex networks.

**B. Principle of Simulated Annealing**

Annealing is a physical process: after the metal object is heated to a certain temperature, all its molecules move freely in the state space $D$. With the gradually decreasing of temperature, these molecules gradually stop at different states. Statistical mechanics research shows that at the temperature of $T$, the probability of molecular objects stays in the state of $r$ to meet the Boltzmann distribution [19]:

$$\Pr[E = E(r)] = \frac{1}{Z(T)} \exp(-\frac{E(r)}{k_B T}). \quad (7)$$

where $E$ is for random variables of the molecular energy, $E(r)$ is for the energy of the molecules at the state of $r$, $T$ is for the temperature, $k_B$ is for the Boltzmann constant, $Z(T)$ is for the normalization factor.

According to the above principles, the application of simulated annealing needs to meet two conditions:

1. The initial temperature of $T = T_0$ is large enough, making the probability molecular stay at any state is similar.

2. When the temperature dropped to 0, all molecules will stay in the lowest energy state with probability 1.

The simulated annealing algorithm is the expansion of local search algorithm. It first generates an initial solution as the current solution. What make it unlike local search algorithm is that it not only searches the minimum cost in the neighborhood of state, but also chooses a non-local optimal solution with probability $P(T)$, makes the solution for the current solution and then continues to repeat, for jumping out of the local optimal state. It is a global optimal algorithm. During the repeat of the process, the probability of choosing non-local optimal solution decrease with the decreasing of temperature, until it chooses the local optimal solution when $T=0$. Metropolis proposed the earliest idea of simulated annealing algorithm in 1953, and Kirkpatrick successfully applied it to the combinatorial optimization problems in 1983 [20].

**IV. ALGORITHM**

According to the characteristics of SDN, local area networks have a specified transparency while the QoS routing has the integrity, so we need the overall consideration. This paper proposes a fully connected method to get the whole network virtual topology with the following two steps: the first step, extracts the boundary nodes in each local area network, and use Dijkstra algorithm to calculate the shortest path between each boundary node to form the fully connected structure; secondly, use the link diffusion method [21] to send the full connectivity information to the around local area network until the control module of each local area network masters the full connectivity structure of all the other local area networks, and then generates the whole network virtual topology. As shown in Figure 1.

![Figure 1. Generation of the whole network virtual topology](image-url)
solution and introduces simulated annealing, and then
iterates the initial solution until find the feasible path or
exceed the acceptable maximum number of iterations with
corresponding decreasing in temperature.

According to the whole network virtual topology, which
obtained by the above methods, we designed
HSA_MCP algorithm for the multi-constrained QoS
routing problem in SDN by basing on the simulated
annealing algorithm and nonlinear energy function.

HSA_MCP( G(V, E) , s, t, c, T0, grad, I)
T=T0;
Init_Dijkstra(G, s);
Init_Dijkstra(G, t);
Simplify_Graphic (G(V, E) );
Init_Dijkstra(G, s);
For (i=0; i<I; i++)
    HSA_Dijkstra(G, t, T);
    If ( (g(u)+r(u)>K)
        Remove node u;
        If (g(u)+r(u)>K)
            For each node u in G
                g(u)=d_i(u)
            
    T=T/grad;
    Return failure;

Init_Dijkstra(G, root)
SPT=[root];
NB=[root’s neighbors];
While NB is not Empty
    u=Cheapest(NB);
    AddNode(SPT, u, NB);
    For each node v in u’s neighbors
        if v is not in SPT
            Relax(u, v);

HSA_Dijkstra(G, root)
SPT=[root];
NB=[root’s neighbors];
While NB is not Empty
    u=HSA_Cheapest(NB, T);
    AddNode(SPT, u, NB);
    For each node v in u’s neighbors
        if v is not in SPT
            HSA_Relax(u, v);
this function use Relax(u, v) to do the relaxation for the neighbor. When we conduct the root=t & root=s situations to mark, we separately use the condition of positive relaxation and reverse relaxation to mark the corresponding conditions.

Simplify_Graphic function. The function is used to judge each node in the graph according to the markers we have. When g(u)+r(u)>K, the node u will not in the feasible paths. Thus, we needn’t think about the node and can delate the node and relative link path from the graph to decrease the scale of topology and shrink the search space.

HSA_Dijkstra function. The operational procedure of this function is similar to Dijkstra function. The difference of these two functions is that this function uses the simulated annealing method, which chooses a node from NB first and add to SPT, and relax the near node by using the HSA_Relax (u,v) function to provide the relaxation conditions. And then, save the annealing marker data to gk (u) for the next annealing marker.

HSA_Cheapest function. This function embodies the core principle of simulated annealing: select non-local optimal node with certain probability, and then, with the decreasing of T temperature, the probability tends to 0.

Now, for the figure 2, we will discuss this algorithm clearly. Let’s assume the source node is 12, the destination node is 43, and the constraints: \( c_1 = 12 \), \( c_2 = 9 \).

![Figure 2. The whole network virtual topology](image)

\[ g(12) = \frac{14}{36}, g(11) = \frac{14}{36}, g(11) = 2, \quad g_2(11) = 2, \quad \pi_{\sigma(11)} = 12. \]

\[ g(23) = \frac{13}{36}, g_2(23) = 1, \quad \pi_{\sigma(23)} = 12. \]

\[ g(31) = \frac{20}{36}, g_2(31) = 4, \quad g_2(31) = 2, \quad \pi_{\sigma(31)} = 12. \]

Then, we will choose the node in NB with the minimum positive linear marker. So it is node 23. Put the node 23 in SPT.

The neighbor nodes of 23 which not in NB are 21, 22, and 24. For the node 21:

\[ g_1(23) + w_i (23,21) + g_2(23) + w_i (23,21) = \frac{26}{9}. \]

\[ g(21) = \frac{18}{36}, \quad g_2(21) = \frac{18}{36}, \quad \pi_{\sigma(21)} = 12. \]

\[ g(22) = \frac{20}{36}, \quad g_2(22) = 4, \quad g_2(22) = 2, \quad \pi_{\sigma(22)} = 23. \]

The markers of node 24 could be calculated in the same way. Finally, put these nodes in NB.

Now, the NB is not empty, so, we can do the same thing as described above.

Init_Dijkstra(G, t). The process of reverse marking is similar to the positive marking described above, with the root node is t and the markers are reverse markers.

After the positive mark and reverse mark, every node in this graph has the positive markers and reverse markers. The result is listed in the table 2 below.

<table>
<thead>
<tr>
<th>Node</th>
<th>11</th>
<th>12</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>31</th>
<th>32</th>
<th>33</th>
<th>41</th>
<th>43</th>
</tr>
</thead>
<tbody>
<tr>
<td>( g(u) )</td>
<td>( \frac{14}{36} )</td>
<td>0</td>
<td>( \frac{18}{36} )</td>
<td>( \frac{20}{36} )</td>
<td>( \frac{13}{36} )</td>
<td>( \frac{30}{36} )</td>
<td>( \frac{20}{36} )</td>
<td>( \frac{27}{36} )</td>
<td>( \frac{30}{36} )</td>
<td>( \frac{58}{36} )</td>
<td>( \frac{71}{36} )</td>
</tr>
<tr>
<td>( \pi_{\sigma(t)} )</td>
<td>12</td>
<td>—</td>
<td>12</td>
<td>23</td>
<td>12</td>
<td>23</td>
<td>12</td>
<td>31</td>
<td>31</td>
<td>32</td>
<td>41</td>
</tr>
<tr>
<td>( r(u) )</td>
<td>( \frac{85}{36} )</td>
<td>( \frac{71}{36} )</td>
<td>( \frac{70}{36} )</td>
<td>( \frac{56}{36} )</td>
<td>( \frac{59}{36} )</td>
<td>( \frac{42}{36} )</td>
<td>( \frac{51}{36} )</td>
<td>( \frac{44}{36} )</td>
<td>( \frac{61}{36} )</td>
<td>( \frac{13}{36} )</td>
<td>0</td>
</tr>
<tr>
<td>( \pi_{\sigma(s)} )</td>
<td>( \frac{12}{36} )</td>
<td>( \frac{24}{36} )</td>
<td>41</td>
<td>24</td>
<td>24</td>
<td>43</td>
<td>32</td>
<td>41</td>
<td>31</td>
<td>43</td>
<td>—</td>
</tr>
<tr>
<td>( \text{cost}(u) )</td>
<td>( \frac{99}{36} )</td>
<td>( \frac{71}{36} )</td>
<td>88</td>
<td>76</td>
<td>76</td>
<td>72</td>
<td>71</td>
<td>71</td>
<td>91</td>
<td>71</td>
<td>71</td>
</tr>
</tbody>
</table>
Simplify_Graphic function. For every node in this graph, we now can calculate the value of cost(u). For example: \( \cos{\theta(22)} = r(22) + g(22) = \frac{19}{9} > 2 \), so the node 22 must not on the path meeting the constraint conditions. By this way, we can delete all the nodes not meet the request and have a simplified graph like figure 3.

For the node 24:

\[
\begin{align*}
Z(24) &= \max \left\{ \frac{d_1(24) + g_1(24)}{c_1} - g, \frac{d_2(24) + g_2(24)}{c_2} - g \right\} = 0, \\
E(41) &= \frac{1}{12}, \\
\end{align*}
\]

Then, we will use the HSA_Cheapest(NB,T) function to choose the annealing node.

For the all nodes in NB:

\[
\begin{align*}
Z(41) &= \frac{13}{12}. \text{ So the value of } g^* \text{ is } 1, \\
E(41) &= \frac{1}{12}. \\
\end{align*}
\]

The value of the normal factor is:

\[
Z = \left[ -\frac{E(24)}{c_1} \right] + \left[ -\frac{E(41)}{c_2} \right] = 1.99 .
\]

We get a random number \( x=0.4 \) with \( 0 \) to \( 1 \).

Choose a node in NB for example 41 to do the calculation: \( \text{sum} = \text{sum} + \frac{E(41)}{Z} = 0.49 > x \), so there we choose node 41 as the annealing node.

Put the node 41 in SPT, and for the neighbor nodes of 41 which not in NB:

\[
\begin{align*}
E(41,24) &= \frac{d_1(24) + g_1(24) + 3}{c_1} - g, \\
E(41,43) &= \frac{d_2(43) + g_2(43) + 3}{c_2} - g, \\
E(41,32) &= \frac{d_3(32) + g_3(32) + 3}{c_3} - g, \\
E(41,13) &= \frac{d_4(13) + g_4(13) + 3}{c_4} - g. \\
\end{align*}
\]

Now, the NB is not empty, so, we can do the same thing as described above.

In this way, we finally can get a path from the node s to the node t. while the path dost not meet the constraint conditions, we can do the annealing mark form the other direction until the path meet the constraints or exceed the acceptable maximum number of iterations.

V. SIMULATION

In the simulation experiment, we use the gt-itm application [22] to generate the SDN network topology and each transit domain corresponds to one local area network in SDN. This experiment evaluates the performance of the HSA_MCP algorithm from the running time and the success rate (SR) of routing calculation. In the simulation experiment, we will compare the performance of the HSA_MCP algorithm with the best algorithm H_MCOP and the general SA_MCP algorithm.

Because the size of network associates with the number of the transit domain and the size of the transit domain, so the simulation experiment generate the topology graphs in the following nodes 100, 200, 300, 400, 600 with the below two ways: 1. change the number of the transit domain while the size of the transit domain keeps constant; 2. change the size of the transit domain while the number of the transit domain keeps constant.

In the simulation experiment, we randomly select 100 pair of \( s-t \) nodes in the topology and calculate the feasible path by HSA_MCP algorithm to record the running time and the result and then, get the success ratio by calculating the running time and the ratio of the success times and the total times.

Because the performance of QoS routing algorithm is affected by the parameter weight ratio, therefore the simulation experiment couples the \( K=4 \) QoS constraint parameters as one parameter according to a certain weight ratio:

\[
g(u,v) = \sum_{k=1}^{K} a_k w_k(u,v). \quad (8)
\]

And then we use Dijkstra algorithm to find the shortest path and use the QoS parameters of the path as the constraint conditions. This simulation experiment design totally 3 weight ratio of 1:0:0:0, 0:1:0:0 and 1:1:1:1.

Fig. 4 shows the relationship of the HSA_MCP success rate and the parameter weight ratio with the number of QoS constraints \( K=4 \), the maximum number of iterations \( I=2 \). In the figure 4, axis is the network size and the vertical axis is the success rate of the HSA_MCP algorithm. As you can see from Fig. 4, with the gradually increase of the network size, the success rate with the
weight ratio of 1:0:0:0 and 1:1:1:1 closes to 100% and relatively stable, while the success rate with the weight ratio of 0:1:0:0 decrease quickly. This is mainly because the energy function algorithm is calculated by the weight ratio of 1:1:1:1. When the value of the weight ratio changed, the performance of the algorithm will be changed, too.

The performance of the algorithm is affected by the influence of the parameter weight ratio. Therefore, unless otherwise specified, the following parameter weight ratio we use will be 1:1:1:1.

The relationship of the network and the time to generate the whole network virtual topology as below:

![Figure 4](image1.png)

**Figure 4.** The success rate in different parameter weight ratio

![Figure 5](image2.png)

**Figure 5.** The time to generate the whole network virtual topology

![Figure 6](image3.png)

**Figure 6.** The running time of algorithms with 100 nodes

The running time of algorithms with 100 nodes

![Figure 7](image4.png)

**Figure 7.** The running time of algorithms with 400 nodes

![Figure 8](image5.png)

**Figure 8.** The running time of algorithms with 600 nodes

Fig. 6, 7 and 8 respectively show the running time of H_MCOP algorithm, SA_MCP algorithm and HSA_MCP algorithm with the size of the network N=100, 400, 600, the maximum number of iterations I=1, 2, 3.

By comparison Fig. 6, 7 and 8, we can summarize that the running time from shortest to longest as the following: H_MCOP algorithm HSA_MCP algorithm and SA_MCP algorithm. When the network size is small, the running time of HSA_MCP algorithm and SA_MCP algorithm is close; but when the network size expands gradually, HSA_MCP algorithm gradually appears obvious advantage than SA_MCP, and the running time trends to close the H_MCOP algorithm. This is mainly because the HSA_MCP algorithm has an optimization mechanism for the expansion of the network size and simplifies the network topology before calculation. Optimization mechanism is not obvious with a small network size. When the network size expands gradually, the effect will be more and more obvious. So the HSA_MCP appears the great suitability to the network size. It can maintain good performance with the gradually expansion of network size.

Fig. 6, 7 and 8 also illustrate the relationship of the HSA_MCP running time algorithm and the maximum number of iterations I. It still indicates the number of iterations has a small influence for the algorithm running time, not like the linear relationship directed by theoretical analysis.
Fig. 9, 10, and 11 respectively show the success rate of H_MCOP algorithm, SA_MCP algorithm and HSA_MCP algorithm with the size of the network $N=100, 400, 600$, the maximum number of iterations $I=1, 2, 3$ and the success of SA_MCP & HSA_MCP. Because the success rate of HSA_MCP algorithm is very high with the parameter weight ratio of $1:1:1:1$, so we generate the QoS constraint condition with the weight ratio of $0:1:0:0$.

It can be concluded from Fig. 9, 10, and 11 that the success rate of HSA_MCP algorithm is higher than that of SA_MCP algorithm. When the maximum number of iterations is 2 or above, the success rates of the SA_MCP algorithm and HSA_MCP algorithm will increase correspondingly, and higher than that of H_MCOP algorithm. Because the success rate of the algorithm is very high in the parameter weight ratio $0:1:0:0$ with $I=2$, so in most cases HSA_MCP can achieve higher success rate with $I=2$ and the running time is also quite reasonable as the Fig. 6, 7, and 8 showing.

Through the simulation experiment described above, we can conclude that the HSA_MCP algorithm we proposed can regulate the success rate by adjusting the number of iteration with the comparison of the classical H_MCOP algorithm. When $I=2$ or above, HSA_MCP algorithm can basically guarantee the success rate not less than that of H_MCOP algorithm. While the running time HSA_MCP algorithm need is bigger than H_MCOP algorithm, but when the size of the network increasing, the running time HSA_MCP algorithm need is become approach to H_MCOP algorithm and have advantage compared with SA_MCP algorithm. At the same time, compared with the general SA_MCP algorithm, when the network size expands gradually, HSA_MCP algorithm is better than the general SA_MCP algorithm in running time. In short, when $I=2$, HSA_MCP algorithm shows good performance, and has certain adaptability to the network size.

VI. CONCLUSION

QoS routing problem is a NP problem without a good solution and the QoS routing problem in SDN networks, as a new research field, has few research results. In this paper, we put forward a new kind of QoS routing algorithm based on the simulated annealing and nonlinear energy function, and transplant it to the SDN according to the characteristics of SDN. Compared with the classical H_MCOP algorithm and general SA_MCP algorithm, we can conclude that: HSA_MCP algorithm has better success rate than H_MCOP algorithm and SA_MCP algorithm, and obviously reduction on the running time than SA_MCP algorithm. When $I=2$, HSA_MCP algorithm can show good performance, and have certain adaptability to the network size. This paper only considers the additive parameters and can be extended to consider other types of QoS constraint parameters. In the simulation experiment, we just research on the $K=4$ situation, and later we can also research on the influence of the number of constraint parameters on the performance of the algorithm by changing the value of $K$. In short, there is certainly space for improvement of this algorithm.

REFERENCES


ACKNOWLEDGMENT

This work was supported by the National Basic Research Program of China (973 Program) under Grant No. 2012CB315803.

Lijie Sheng is an associate professor in School of Computer Science, XiDian University. In July 2006, he graduated from Department of Computer Science and Technology, Tsinghua University, and received a doctor's degree in computer science and technology. His research interests include future Internet Architecture and parallel programing.

Zhikun Song received the bachelor degree in computer science and technology from ZhengZhou University in 2013. He is a graduate student in the Department of Computer at XiDian University. His research interests include Software Defined Network, the next generation Internet, Windows programing.

Jianhua Yang received the bachelor degree in computer science and technology from Henan Agricultural University in 2013. He is a graduate student in the Department of Computer at XiDian University. His research interests include Software Defined Network and the future Internet Architecture.