SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2016, Vol. 59 092201:1–092201:12
doi: 10.1007/s11432-015-5364-0

Virtual Strategy QoS routing in satellite networks

Heyu LIU*, Fuchun SUN & Shaoqing WANG

State Key Laboratory of Intelligence Technology and Systems, Department of Computer Science and Technology, Tsinghua University, Beijing 100048*, China*

Received December 23, 2015; accepted March 3, 2016; published online August 23, 2016

Abstract Satellite network users want to unify different satellites to enhance the quality-of-service (QoS) stability of the satellite networks and select feasible paths through different networks to feed different applications. Unfortunately, the state of affairs is that different applications need to use several different application programming interfaces and to design different protocols on how and when to use a specific network. This is troublesome and error-prone as the application programming interfaces vary a lot. In this paper, we design a Virtual Strategy in satellite network based on which a QoS routing service scheme is then proposed. We analyze why applications should use and benefit from Virtual Strategy. This Virtual Strategy is a middleware solution that enables seamless usage of services from different satellite network parts. And then, the supporting QoS routing solution enables the committed QoS services over Virtual Strategy. Finally, we provide a comparison between the previous satellite networks and our work. The simulation results show that our Virtual Strategy QoS routing scheme demonstrates dominated performances under complex architecture.

Keywords satellite network, QoS services, Virtual Strategy, middleware, routing scheme

Citation Liu H Y, Sun F C, Wang S Q. Virtual Strategy QoS routing in satellite networks. Sci China Inf Sci, 2016, 59(9): 092201, doi: 10.1007/s11432-015-5364-0

1 Introduction

We have witnessed a rapid development in satellite network communication technologies during the past decades. Stimulated by the explosive QoS requests, new applications try to leverage this new wireless and wired technologies and endeavor to participate in the resource competition in hybrid networks. Unfortunately, most of them use specified schemes but are suffocated to make it in even the Internet. Also, different applications mostly use socket interface and associated application programming interfaces that differ between programming languages and operating systems. As a consequence, QoS routing scheme is strictly restricted in specified environment without sophisticated interactions between interfaces.

An application that wants to use several QoS schemes for sending and receiving data has to decide which technologies, programming languages, and protocols to opt. The current schemes might be locally connected or remote, which brings yet another problem: how to use, configure, and share a remote device.

In the past few years, several methods have been studied intensively in this field [1–5]. Some papers propose the use of multi-protocols [6] for supporting routing request from different applications in the

* Corresponding author (email: liuheyukicker@163.com)

same network, but it will enhance the complexity of the operation system, while other papers support multi-application programming interfaces [7] which need to be switched by hand.

Regardless of the multi-protocol-based or multi-interface-based strategy, a common view is that a unified strategy control plane provides more efficient network resource utilization over the whole network than when per-application independent control plane is used, especially referring to routing. That is why we introduce the Virtual Strategy [8–11] into the satellite networks.

In this paper, we design a Virtual Strategy first of all. Virtual Strategy is a middleware solution providing a unified interface which applications can use to accommodate different communication technologies to configure protocol properties or to transmit messages. The physical communication device can be remote or locally attached in the current scheme. And then, based on the strategy, a QoS service routing scheme is proposed, which assists to satisfy the QoS service commitments after interface protocols are unified. The use cases of Virtual Strategy QoS routing can be found from video and energy applications, while data messages simply need to reach their destinations regardless of network or application differences. The contributions and benefits of Virtual Strategy QoS routing scheme are to comply with a unified service protocol instead of multi-protocols to eliminate unnecessary interactions.

The rest of the paper is organized as follows. In Section 2, the logical architecture of the satellite network, including the Virtual Strategy middleware is given, as well as the main details of the middleware control mechanism. Section 3 proposes the Virtual Strategy details and designs the mapping algorithm which make the satellite devices into the virtual network resources. Section 4 introduces the QoS service routing scheme–stochastic gradient ascent (SGA) updated swarm intelligence QoS routing scheme. Section 5 presents simulation results and discusses about real satellite implementations. The paper is concluded in Section 6.

2 Overview of logical architecture

Virtual Strategy can be considered as message-oriented middleware. It embeds the physical satellite networks into the virtual satellite networks.

As shown in Figure 1, satellite network virtualization, by means of resource consolidation that maps several virtual instances to one physical resource device, is feasible to save resource in future infrastructure networks. Several satellites are treated as one enhanced logical satellite underlying the physical satellite devices. As shown in Figure 2, after a k-cluster algorithm, the satellites in different orbits and different altitudes would be categorized into different groups according to the application demands.

Applying virtualization of network resource leads to the problem of allocating virtual network demands to physical network resources. The dynamic satellite network will mainly exploit periods of low traffic demands when some routers and interfaces can be switched off by rerouting the traffic to a smaller set of consolidated network equipments with increased utilization.

Figure 3 shows the overview of the Virtual Strategy middleware and the interactions between virtual networks and the physical satellite devices.

This architecture is similar to the virtualization system in terrestrial ground networks. In state-of-the-art virtualization systems, this is often achieved through the use of a software module known as a middleware, which works as an arbiter between a virtual network and underlying physical devices.

The use of the Virtual Strategy middleware brings many advantages, such as satellite power enhanced, device sharing, performance isolation, and even virtual network internetwork security.

Committing a middleware consulting every time a virtual network needs to make a privileged call, however, introduces considerable overhead as the middleware must be brought online to process each request.

3 Virtual Strategy and mapping algorithm

As a middleware, Virtual Strategy provides QoS service for applications. By network interface, or simple interface, a combination of hardware and software can interact with physical satellite network and virtual

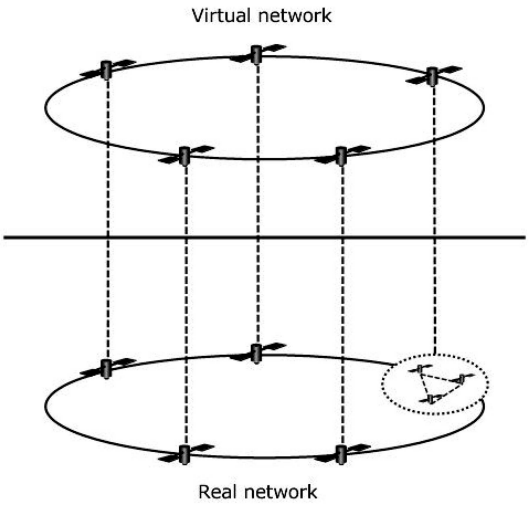

Figure 1 Relation of virtual network and real network.

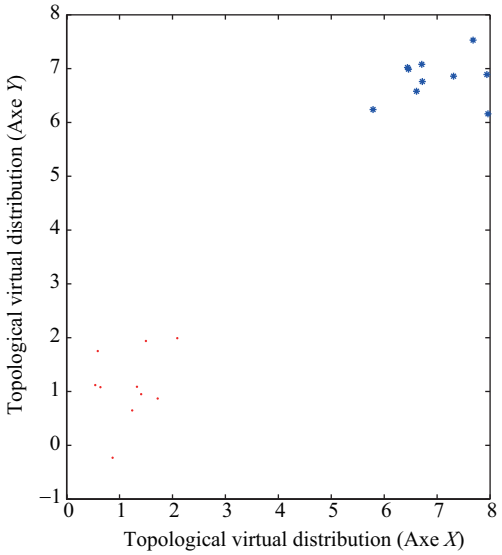

Figure 2 (Color online) Departure of satellites after k-cluster.

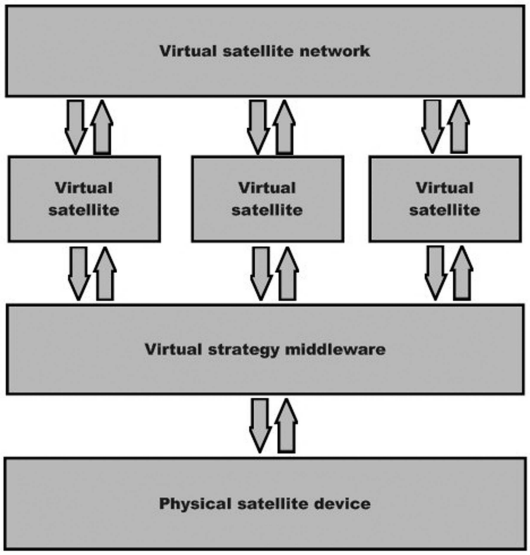

Figure 3 Virtualization-based satellite network architecture.

satellite network to provide the following useful properties.

Address mapping: Each satellite network has its own native address but applications do not need to know about them. Virtual Strategy middleware offers an address mapping between applications and networks native address formats, through which applications only need to focus on their own addresses.

Neighborhood discovery and management: Usually applications need to know where to find the recipient of a message, thus they need to discover and manage their application neighbors. Virtual Strategy middleware can also be used to abstract this tedious task: applications only need to declare that they want to send to this neighbor.

Queue management: Due to satellite network characteristics, some messages may need enqueue or even retransmission. Virtual Strategy middleware abstracts this process from applications and only needs to define if a message needs to be transmitted reliably or unreliably.

Network switching at failure: Communication networks are vulnerable. If an application relies only on a specific network, its operation becomes nonrobust too. If there are several networks available, Virtual

Strategy middleware can help to switch smoothly to an alternative network.

Traffic control: Virtual Strategy middleware can deploy its own traffic control mechanism that obtains network forwarding capabilities without saturating any intranetwork.

Network interface configuration: Virtual Strategy middleware can conduct logic-based configuration on network interface by making adjustments to the parameters which are convinced configurable by interfaces.

The methods to fully realize virtualization fall into some categories, as described below.

3.1 Virtual satellite network model

As mentioned, virtual satellite network is defined as a combination of virtual satellite nodes and virtual QoS links, but not necessarily physically established. Virtual satellite nodes are logical nodes, representing some neighbor satellite devices in the same region. Virtual QoS links represent the potentiality to set up a virtual network with physical network link state.

A virtual QoS link can be either fully or partially determined before establishment. When fully determined, the path of underlying link state is entirely precomputed and stored. When partially determined, only part of the path information is preknown, allowing the flexibility to establish associated link states.

The main advantage of using virtual satellite nodes and links instead of pre-established physical network in virtual network configuration is that the associated nodes and links occupy resource that may not be optimally configured. This may preclude using such optical resources to set up other, eventually more appropriate, virtual networks.

3.1.1 *Original network model*

Original network is denoted as a weighted undirected graph $G^S = (N^S, L^S, C_N^S, C_L^S)$, in which N^S is the set of original nodes and L^S is the set of original links. Nodes are equipped with a capacity of C_N^S and the C_L^S represents the bandwidth of the original links. We use $\boldsymbol{P}^S$ to denote the set of original paths.

3.1.2 *Virtual network model*

Like original network, virtual network request could also be characterized by a four-tuple. $G^V = (N^V, L^V, C_N^V, C_L^V)$, where N^V is denoted as the set of nodes and L^V represents the set of QoS links in the virtual network. C_N^V and C_L^V represent the node and link constraints, respectively.

3.2 Mapping algorithm

Mapping algorithm established a bond between the physical network and the virtual network. It includes two steps: node assignment and QoS link assignment, and the details are listed below.

3.2.1 *Node assignment*

Virtual node in the virtual network request is mapped to several physical nodes which have more resources available than virtual node request,

$$M_N : n^V \to n^S (n^V \in N^V, n^S \in N^S) \ \text{ s.t. } C_N(n^S) > C_N(n^V), \tag{1}$$

where n^S is a physical node and n^V is its mapped virtual node. In this paper, each physical node can host only one virtual node in virtual network request.

3.2.2 *QoS link assignment*

Each virtual QoS link in the virtual network is mapped to a physical path,

$$M_L : l^V \to p^S (l^V \in L^V, p^S \in P^S) \ \text{ s.t. } C_L(l^S) > C_L(l^V)(l^S \in p^S). \tag{2}$$

Algorithm 1 The mapping algorithm

```
Input: The physical satellite network;
Output: The virtual satellite network;
set physical node array N_S[N];
set physical QoS link array L_S[X];
set virtual node array N_V[M];
set virtual QoS link array L_V[Y];
while k-cluster algorithm do
   for each i = 1 to M do
      N_V[i] = N_S[k];
   end for
end while
for each k = 1 to Y do
   for each i = 1 to M do
      for each j = 1 to M do
         link N_V[i] and N_V[j];
      end for
   end for
   L_V[K] = link L_S[i] and L_S[j];
end for
```

3.2.3 *Mapping algorithm procedure*

The pseudo-code of the mapping algorithm is described in Algorithm 1.

4 QoS routing scheme over Virtual Strategy

After the Virtual Strategy having laid down the fundamentals of a unified protocol environment to neglect different networks or application interfaces, it is naturally essential to propose an appropriate routing scheme for those committed QoS requests. Unfortunately, all the current schemes in literature are based on some specified application interfaces without compromising the virtualization and mapping processes in the above strategy.

Swarm intelligence is widely applied in many research areas for its outstanding adaptability to multi-handover environments. However, as far as we know, no swarm intelligence-related routing scheme has been proposed to cast light on the routing problems under complex conditions. Thus, in this section, a SGA (stochastic gradient ascent) updated swarm intelligence QoS routing scheme is illustrated in detail.

4.1 Model of the routing scheme

In this part, we briefly introduce the model of a general swarm intelligence routing scheme. Swarm intelligence is a umbrella name of some intelligent methods inspired by the behaviors of biotic populations [12], including ant colony intelligence, bee colony intelligence, fish colony intelligence, and so on. To keep the commonality, general solutions instead of a specific one are used in this paper.

Under Virtual Strategy, the differences between networks or interfaces are contemporarily hidden in the topology. A topology is abstracted into a weighted construction graph, $G = (N, L, \boldsymbol{P})$, where N represents nodes set, L represents QoS links set, and $\boldsymbol{P}$ is the weight vector of QoS links, for example, the pheromone in ant colony intelligence or the preferences toward QoS links after walking dances among bee agents in bee intelligence. $\boldsymbol{P}$ is comprised of $p_{i,j}$, denoting the weight of link (i, j). Each solution to the swarm intelligence routing can be expressed as $s = \langle n_s, n_1, n_2, ..., n_k, n_d \rangle$, in which $n_i \in N$, $k+2 \leqslant |N|$. s should satisfy the parameter restrictions (denoted as ψ) suggested by the QoS service.

At the initial stage, individual agents are randomly generated in all the source ends to route toward destination ends. The agents route randomly according to probabilities decided by $\boldsymbol{P}$. Agents immediately die whenever ψ is unsatisfied. The routing process is simplified as follows:

Step 1. Agents are generated in source end n_s to route toward destination end n_d.

Step 2. For a sequence $\langle n_s, n_1, n_2, ..., n_k\rangle$, if $n_k \neq n_d$, then $\forall n_j (n_k, n_j) \in L$ and $\langle n_s, n_1, n_2, ..., n_j\rangle$ satisfies ψ, individual agent selects the traversing QoS link according to

$$P(n_{k+1} = n_j \,|\boldsymbol{P}) = \frac{F(p_{k,j})}{\sum_j F(p_{k,j})}. \tag{3}$$

Step 3. If $n_{k+1} = n_d$, the process terminates, and $s = \langle n_s, n_1, n_2, ..., n_k, n_d\rangle$ is added to the solution space S. Else, the process returns to Step 2.

The function $F()$ in Step 2 directly determines the considered factors together with their weights on selecting the QoS link. Each intelligence equips its own tuned function. Ant colony, for example, sets $F(p_{k,j}) = p_{k,j}$ to reduce complexities of routing; while the fish colony uses $F(p_{k,j}) = {p_{k,j}}^{\alpha}{\theta_{k,j}}^{\beta}$ to upgrade the effect to end nodes from the number of agents.

The elementary idea of swarm intelligence is to achieve the comprehensive high intelligence on swarm level through the interaction and coordination among individual agents. Thus, a suitable interaction and coordination mechanism is vital to the intelligence performances. In QoS routing scheme, the mechanism is specified to the updating probabilities on selecting links. Through the iterated updating mechanism, links with higher quality are allocated higher weights, which contribute to the possibility of comprising optimal path. A general form of iterated mechanism could be expressed as

$$\forall s \in S_t, \quad p_{i,j} = \begin{cases} (1-\lambda)(p_{i,j} + Q_c(s\,|S_1, S_2, ..., S_t)) & (i,j) \in s, \\ (1-\lambda)p_{i,j} & \text{others}, \end{cases} \tag{4}$$

where S_t is the effective path set at the ith iteration and quality function $Q_c()$ acts as a feedback update to a QoS link based on the cost function C_0 of a path. λ is volatilization constant to control the changing rate of link selection probability along with time.

As the swarm intelligence derives from the observation of biological phenomena, the conventional iterated updating mechanisms are unavoidably affiliated with some biological characteristic behaviors, which might deteriorate the iteration rates to some degree. In the next part, we introduce the SGA method to accelerate the process systematically.

4.2 Stochastic gradient ascent

In this part, we introduce the SGA method.

SGA method is a model-based updating algorithm, which is equipped with a storage space to record the historical iteration data and optimize the probability model.

A optimization problem with solution set S and cost function $C()$ could be expressed as $s^* = \arg\min_{s\in S} C(s)$. Ref. [13] proves that if the probability space model $\Theta = \{K_P() \,|\boldsymbol{P} \in P^T\}$ is an N-dimensional parameter space ($P^T \subset R^n$) and could be parameterized completely smoothly by an N-dimensional parameter vector $\boldsymbol{P}$, then the original optimization problem equals to

$$\boldsymbol{P}^* = \arg\max_{\boldsymbol{P}} \varepsilon(\boldsymbol{P}), \text{ in which } \varepsilon(\boldsymbol{P}) = E_{\boldsymbol{P}} Q_c(s)\ E_{\boldsymbol{P}} = \sum_s K_{\boldsymbol{P}}(s). \tag{5}$$

$\boldsymbol{P}^0$, the given vector, is used to initiate the iteration. The gradient ascent indicates

$$\boldsymbol{P}^{t+1} = \boldsymbol{P}^t + \theta \nabla \varepsilon(\boldsymbol{P}^t), \tag{6}$$

where θ is the step length in each iteration. The composition of gradient could be further deduced:

$$\begin{aligned} \nabla\varepsilon(\boldsymbol{P}) =& \nabla E_{\boldsymbol{P}} Q_c(s) = \nabla \sum_s K_{\boldsymbol{P}}(s) Q_c(s) \\ =& \sum_s Q_c(s) \nabla K_{\boldsymbol{P}}(s) = \sum_s Q_c(s) K_{\boldsymbol{P}}(s) \frac{\nabla K_{\boldsymbol{P}}(s)}{K_{\boldsymbol{P}}(s)} \\ =& \sum_s Q_c(s) K_{\boldsymbol{P}}(s) \nabla \ln K_{\boldsymbol{P}}(s) = E_{\boldsymbol{P}} Q_c(s) \nabla \ln K_{\boldsymbol{P}}(s). \end{aligned} \tag{7}$$

The above expression cannot be simply transplanted to iteration process for its method to calculate $E_{\boldsymbol{P}}$ implies a complete search in the whole searching space. In practical terms, a stochastic option is often adopted. It translates $E_{\boldsymbol{P}}$ to a empirical mean value $\boldsymbol{P}^{t+1}$, decided by a sample value S_t in last iteration,

$$\boldsymbol{P}^{t+1} = \boldsymbol{P}^t + \theta \sum_{s \in S_t} Q_c(s) \nabla \ln K_{\boldsymbol{P}^t}(s). \tag{8}$$

4.3 SGA updated swarm intelligence routing scheme

4.3.1 *SGA update*

To exploit the SGA algorithm in QoS routing scheme, some details that are required are further studied.

When $S = \langle n_s, n_1, n_2, ..., n_k, n_d \rangle$, the $K_{\boldsymbol{P}}(s)$ in Eq. (7) could be rewritten as

$$\begin{aligned} K_{\boldsymbol{P}}(S) &= \prod_{i=1}^{|S|-1} K_{\boldsymbol{P}}(n_{i+1}|\text{pros}_i(S)), \\ \nabla \ln K_{\boldsymbol{P}}(S) &= \sum_{i=1}^{|S|-1} \nabla \ln K_{\boldsymbol{P}}(n_{i+1}|\text{pros}_i(S)). \end{aligned} \tag{9}$$

Function pros() represents the previous i elements of S.

Any QoS link(i,j) on the weighted construction graph falls into three categories according to its positional relationship with Path S:

Link(i,j) resides on Path S:

$$\begin{aligned} &\frac{\partial}{\partial p_{i,j}} \{\ln K_{\boldsymbol{P}}(n_{i+1}|\text{pros}_i(S))\} \\ &= \frac{\partial}{\partial p_{i,j}} \left\{ \ln \frac{F(p_{i,j})}{\sum_{i=n_k,(i,j)\in\psi} F(p_{i,j})} \right\} = \frac{\partial}{\partial p_{i,j}} \left\{ \ln F(p_{i,j}) - \ln \sum_{i=n_k,(i,j)\in\psi} F(p_{i,j}) \right\} \\ &= \frac{F'(p_{i,j})}{F(p_{i,j})} - \frac{F'(p_{i,j})}{\sum_{i=n_k,(i,j)\in\psi} F(p_{i,j})} = \{1 - K_{\boldsymbol{P}}(j|\text{pros}_i(S))\} \frac{F'(p_{i,j})}{F(p_{i,j})}. \end{aligned} \tag{10}$$

Link(i,j) intersects with Path S:

$$\begin{aligned} &\frac{\partial}{\partial p_{i,j}} \{\ln K_{\boldsymbol{P}}(n_{i+1}|\text{pros}_i(S))\} \\ &= \frac{\partial}{\partial p_{i,j}} \left\{ \ln \frac{F(p_{i,n_{i+1}})}{\sum_{i=n_k,(i,j)\in\psi} F(p_{i,j})} \right\} = \frac{\partial}{\partial p_{i,j}} \left\{ \ln F(p_{i,n_{i+1}}) - \ln \sum_{i=n_k,(i,j)\in\psi} F(p_{i,j}) \right\} \\ &= 0 - \frac{F'(p_{i,j})}{\sum_{i=n_k,(i,j)\in\psi} F(p_{i,j})} = -K_{\boldsymbol{P}}(j|\text{pros}_i(S)) \frac{F'(p_{i,j})}{F(p_{i,j})}. \end{aligned} \tag{11}$$

Other situations:

$$\frac{\partial}{\partial p_{i,j}} \{\ln K_{\boldsymbol{P}}(n_{i+1}|\text{pros}_i(S))\} = 0. \tag{12}$$

The weights of corresponding QoS links in the weighted construction graph get updated using Eq. (10)–(12) in each iteration.

4.3.2 *Routing scheme*

With the three different updating methods deduced above, SGA updated swarm intelligence QoS routing scheme over Virtual Strategy is described as follows:

Step 1. Start of a Virtual Strategy, all QoS links are initialized, a commonly used initialization distribution is uniform distribution, that is for each link $p_{i,j}^0$ on $\boldsymbol{P}^0$:

$$p_{i,j}^0 = \begin{cases} \dfrac{1}{|L|}, \text{ for } (i,j) \in L, \\ 0, \text{ otherwise.} \end{cases} \tag{13}$$

Meanwhile, the agents are randomly generated in all source ends.

Step 2. Judging whether an agent is still alive, if it has ran out of its lifetime, the agent deceases; or it uses Eq. (3) to select a QoS link and enters the next hop.

Step 3. If the current node is the destination end d, the algorithm shifts to Step 4; or it goes back to Step 2.

Step 4. Start of the iteration, for a constructed Path S, algorithm conducts SGA update according to deduced equations. Different links adopt different update methods.

(i) Reside on the path

$$\begin{aligned} \Delta \boldsymbol{P}_{\text{on-s}}^{t+1} =& \theta \sum_{s \in S_t} Q_c(s) \nabla \ln K_{\boldsymbol{P}^t}(s) \\ =& \theta \sum_{s \in S_t} \sum_{i=1}^{|s|} Q_c(s) \nabla \ln K_{\boldsymbol{P}^t}(n_{i+1} | \text{pros}_i(s)) \\ =& \theta \sum_{s \in S_t} \sum_{i=1}^{|s|} (1 - K_{\boldsymbol{P}^t}(i+1 | \text{pros}_i(s))) Q_c(s) \frac{F'(p_{i,i+1}^t)}{F(p_{i,i+1}^t)}. \end{aligned} \tag{14}$$

(ii) Intersect with the path

$$\begin{aligned} \Delta \boldsymbol{P}_{\text{cross-s}}^{t+1} =& \theta \sum_{s \in S_t} Q_c(s) \nabla \ln K_{\boldsymbol{P}^t}(s) \\ =& \theta \sum_{s \in S_t} \sum_{i=1}^{|s|} Q_c(s) \nabla \ln K_{\boldsymbol{P}^t}(n_{i+1} | \text{pros}_i(s)) \\ =& -\theta \sum_{s \in S_t} \sum_{i=1}^{|s|} \sum_{(i,j) \in \psi} Q_c(s) K_{\boldsymbol{P}^t}(j | \text{pros}_i(s)) \frac{F'(p_{i,j}^t)}{F(p_{i,j}^t)}. \end{aligned} \tag{15}$$

Step 5. Before every round of iteration ends, all links are uniformly updated as $p_{i,j}^{t+1} = (1-\lambda) p_{i,j}^{t+1}$.

Essentially speaking, the SGA update tries to reward the high-quality links ($\Delta \boldsymbol{P}_{\text{on-s}}^{t+1} > 0$) while punishing the low-quality links ($\Delta \boldsymbol{P}_{\text{cross-s}}^{t+1} < 0$). Meanwhile, a volatility mechanism is also introduced to avoid local optimum. QoS links with higher qualities ($p_{i,i+1}^t$) are allocated higher and higher probabilities to be selected and finally converge to constitute the optimal path.

5 Simulation and analysis

The best way to show the difference in overhead between QoS routing on original satellite networks and that mechanism on satellite networks with Virtual Strategy middleware is to design experiments measuring the overhead of sending and receiving on different systems.

5.1 Performance of Virtual Strategy

We construct a satellite network environment using OPNET software to simulate the satellite topology and compile a Virtual Strategy middleware from source and install it in our test system. Our test system was a modern mid-range PC with an Intel Core 2 Q9500 quad core processor running at 2.83 GHz. The PC was equipped with 4 GB of 1333 MHz DDR-3 SDRAM and a 320 GB 7200 RPM hard drive with

Table 1 system performance with 10 Mb/s traffic

Elements	Original satellite network	Satellite network with Virtual Strategy
Processor cycles (Mb/s)	50.6	94.7
LLC (Mb/s)	2.1	3.7
Interrupt requests (/s)	2400	4900
Context switches (/s)	449	449

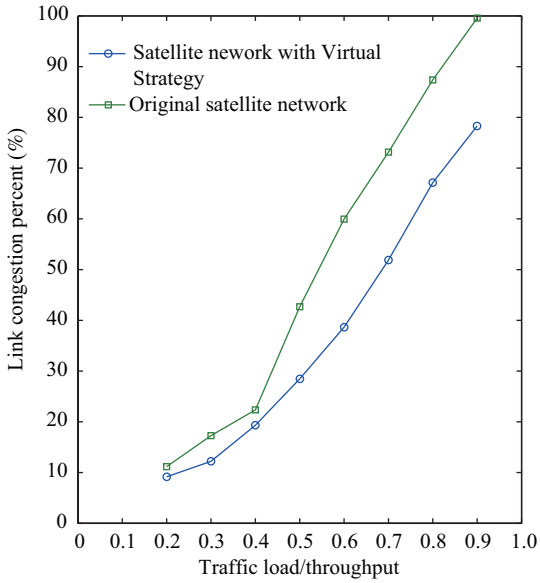

Figure 4 (Color online) Link congestion percents with traffic load increasing.

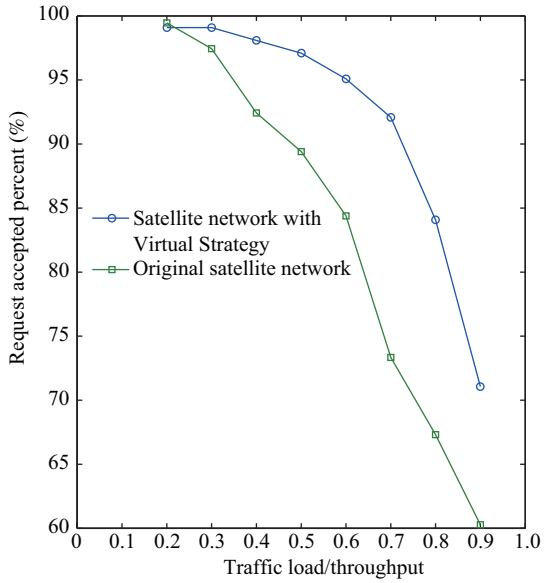

Figure 5 (Color online) Request accepted percents with traffic load increasing.

16 MB cache. The physical network interface is a 1000 Mb/s Broadcom Ethernet adapter attached to the PCI-E (peripheral component interconnection express) bus.

To demonstrate the superiority of Virtual Strategy over conventional network, we use the hardware performance analysis tool Perf to collect system-level statistics such as processor cycles consumed, the processor's last level cache (LLC), interrupt requests, and processor context switches. The results for the receiving experiment are given in Table 1.

Obviously, in the receiving experiment, satellite network with Virtual Strategy takes nearly 2 times more cycles to deliver the packets to the original satellite network.

Figures 4 and 5 are obtained results with regard to two different metrics: link congestion percent and request accepted with the load increasing. From these results, we can extract the following observations:

1. Given the consistent background traffic, satellite network with Virtual Strategy can provide more link resource QoS routing than original satellite network.

2. Given the consistent background traffic, satellite network with Virtual Strategy can support more application QoS request and give better QoS service performances than original satellite network.

5.2 Performance of QoS routing scheme

Unlike the experiment above, to give a thorough description of corresponding QoS routing scheme requires the details of real satellite network topologies and background traffic data.

The constellation model contains one Walker-Star polar subconstellation with 50 satellites evenly distributed in five polar orbits, together with two high-slope long-radius elliptical orbit subconstellation (Hipparcos). The height of each satellite is 1046 km for Walker and 9172 km for Hipparcos. Besides, the ISLs (inter-satellite links) are full-duplex links with 160 Mbps bandwidth and the buffers of satellites in both constellations are 5 MB; each packet is fixed to 1000 bytes. Owing to limited space, one could refer

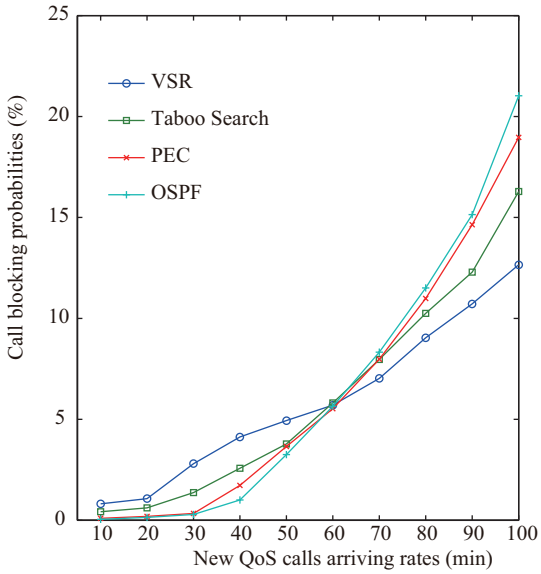

Figure 6 (Color online) The blocking probabilities with new call arriving rates increasing.

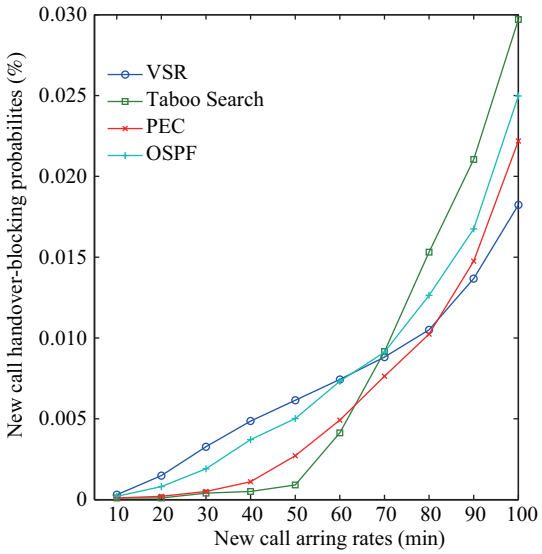

Figure 7 (Color online) The handover-blocking probabilities with new call arriving rates increasing.

to [14] for more details about two subconstellations.

From the aspect of traffic, the background communication traffic is generated through μ pairs of ground users evenly accessing the satellite nodes. New calls are created with a constant possibility, whose service intervals obey negative exponential distribution with 5 min mean. Meanwhile, we also generate background flows whose data rates accord with negative exponential distribution average λ for every call service. The variability of μ and λ provides adjustable background flows.

OSPF routing, Taboo Search routing and , and proper equal constraint (PEC) routing, representing the conventional schemes, act as the comparisons in this experiment.

Before conducting experiments, the implementation of SGA routing scheme is further detailed. First of all, agents are generated randomly in all sources and the QoS links are uniformly distributed. Then, at the end of each iteration, every active agent has to select a possible link with the assistance of Eq. (3). After each iteration ends, marked by the arriving at destination of an agent, Eqs. (14) and (15) are used to update all the links residing on or intersecting with the feasible Path. Besides, to avoid local optimum, all links decay periodically. Through these procedures, the SGA routing is successfully implemented.

Figure 6 depicts the blocking probabilities against the new arriving calls and Figure 7 depicts the new call handover-blocking probabilities under each routing scheme. In both the situations, routing under Virtual Strategy routing (VSR) obviously demonstrates superior performance than other schemes under congest conditions. The main reason is that with the introduction of more network-crossing conversation flows, the conventional protocol negotiation consumes too much link and node resources and sharply deteriorates the QoS service performances. On the other hand, the essence of interaction and coordination in the Virtual Strategy QoS routing eliminates most of the protocol or interface switching, thus enhancing their performances.

However, it should still be mentioned that if most data flows remain in the network under the same protocols or interfaces (just like what happens in Ethernet or network belonging to an Internet service provider), the Virtual Strategy QoS routing scheme is relevant inconvincible, which could be further studied in future.

To demonstrate the superiority of SGA algorithm over traditional QoS routing schemes under the circumstance of Virtual Strategy, the comparisons are also given in Figures 8 and 9. It is not unexpected that SGA provides higher QoS service capabilities than PEC and Taboo Search under the same condition of Virtual Strategy background. From the perspective of routing schemes, the SGA generates iteration

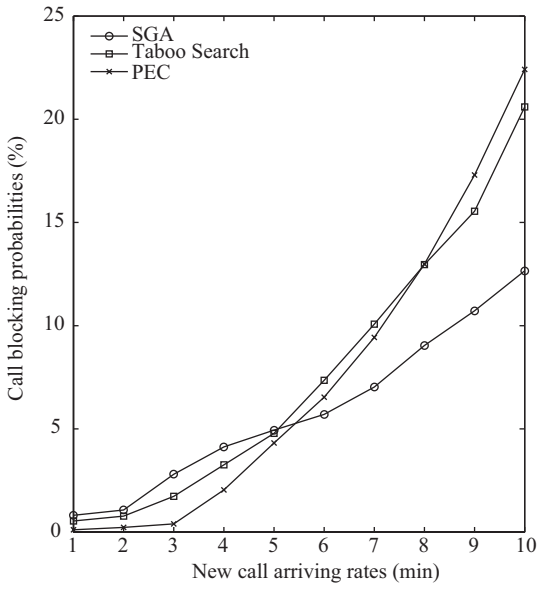

Figure 8 The blocking probabilities with new call arriving rates increasing (under Virtual Strategy).

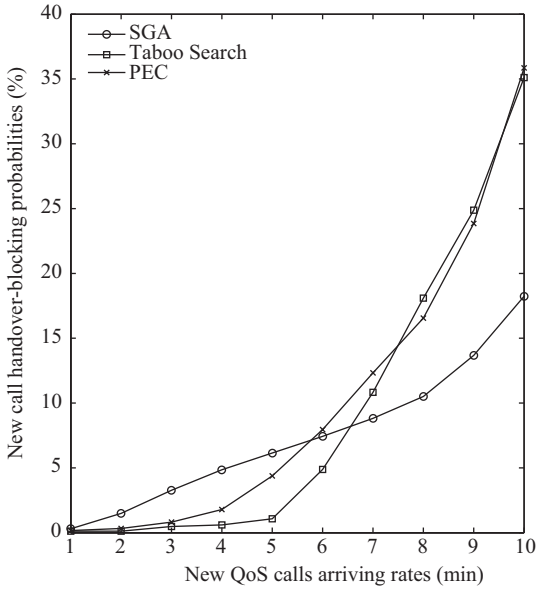

Figure 9 The handover-blocking probabilities with new call arriving rates increasing (under Virtual Strategy).

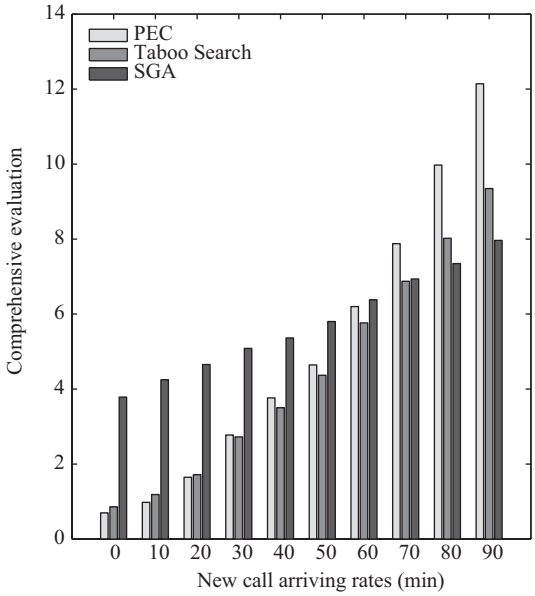

Figure 10 Comprehensive evaluations on routing schemes.

updates more rapidly and efficiently, as illustrated in Eqs. (8) and (3). Also, the differentiation of positional relationship between the QoS links and selected paths eliminated the possibility of local optimum in Taboo Search.

Figure 10 is the result of a comprehensive evaluation on the QoS characteristics under each routing scheme. To give a proper measurement, including all QoS parameters, a evaluation index evaluation valure (EV) is created,

$$EV = \text{delay}/1000 + 100(P_c + \text{PLR}),$$

where delay is the end-to-end delay (ms), P_c are congestion probabilities, and PLR is package loss rate. From the definition of EV, it is obvious that the lower the value of evaluation is, the higher the quality service a scheme could satisfy. From the result, it can be concluded that the SGA routing could achieve

the same order of magnitude in QoS routing performances as the network layer complicates. It should be mentioned that OSPF is not tested for it provides just best-effort services.

6 Conclusion

Handling multiple kinds of applications for different networks can be a tedious task, especially at the stage of routing. Current solutions for this have been concentrated around terrestrial ground networks, such as IP Internet, and we neglected the fact that there are multitude of satellite networks, protocols, or interfaces that do not fit for these methods.

In this paper, we present Virtual Strategy as a middleware that gives applications a possibility to use any satellite in their messaging. We modify the satellite network logic architecture and analyze messaging traffic performance in this new system. We study how the corresponding QoS routing scheme applies to the network under Virtual Strategy. The results show that the Virtual Strategy concept is feasible and can be easily extended, also the QoS routing scheme is superior compared with other schemes over this strategy.

As a future work, we need to design another Virtual Strategy and do some baseline testing with the works of others to see how this Virtual Strategy and other Virtual Strategies are compared. Also, the QoS service performance in the simple domain when the new call arriving rates are not congested requires improvement.

Acknowledgements This paper was supported by National Natural Science Foundation of China (Grant No. 61373144).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Maatta J, Jarvinen R, Luostarinen R, et al. The virtual network system. In: Proceedings of the 2nd International Workshop on Middleware for Pervasive Mobile and Embedded Computing. New York: ACM Press, 2010. 3
2 Saeed A, Habak K, Fouad M, et al. DNIS: a middleware for dynamic multiple network interfaces scheduling. ACM SIGMOBILE Mobile Comput Commun Rev, 2010, 14: 16–18
3 Jarvinen R, Maatta J, Luostarinen R, et al. MICS messaging platform: architecture, design and routing. In: Proceedings of the Military Communications Conference, San Jose, 2010. 1893–1898
4 Bianzino A P, Chaudet C, Rossi D, et al. A survey of green networking research. Commun Surv Tutorials, 2012, 14: 3–20
5 Fischer A, Botero Vega J F, Duelli M, et al. ALEVIN a framework to develop, compare, and analyze virtual network embedding algorithms. Electron Commun EASST, 2011, 37: 1–12
6 Buskirk G A, Santiago R A. US Patent 7 978 606, 2011–7–12
7 Merz S, Quinson M, Rosa C. Simgrid mc: verification support for a multi-api simulation platform. In: Formal Techniques for Distributed Systems. Berlin: Springer, 2011. 274–288
8 Houidi I, Louati W, Ameur W B, et al. Virtual network provisioning across multiple substrate networks. Comput Netw, 2011, 55: 1011–1023
9 Chowdhury M, Rahman M R, Boutaba R. Vineyard: virtual network embedding algorithms with coordinated node and link mapping. IEEE/ACM Trans Netw, 2012, 20: 206–219
10 Liu J. Evaluating standard-based self-virtualizing devices: a performance study on 10 GBE NICS with SR-IOV support. In: Proceedings of IEEE International Symposium on Parallel and Distribution Processing, Atlanta, 2010. 1–12
11 Shea R, Liu J. Understanding the impact of denial of service attacks on virtual machines. In: Proceedings of IEEE the 20th International Workshop on Quality of Service. New Jersey: IEEE Press, 2012. 27
12 Liu H Y, Sun F C. Routing for predictable multi-layered satellite networks. Sci China Inf Sci, 2013, 56: 110102
13 Zlochin M, Birattari M, Meuleau N, et al. Model-based search for combinatorial optimization: a critical survey. Ann Oper Res, 2004, 131: 373–395
14 Montenbruck O, Gill E. Satellite Orbits: Models, Methods and Applications. Berlin: Springer Science & Business Media, 2012. 153–155