Framework for End-to-End Optimal Traffic Control Law Based on Overlay Mesh

Chunyu Liu and Ke Xu

Abstract: **Along with the development of network, more and more functions and services are required by users, while traditional network fails to support all of them. Although overlay is a good solution to some demands, using them in an efficient, scalable way is still a problem. This paper puts forward a framework on how to construct an efficient, scalable overlay mesh in real network. Main differences between other overlays and ours are that our overlay mesh pocesses some nice features including class-of-service (CoS) and traffic engineering (TE). It embeds the *end-to-end optimal traffic control law* which can distribute traffic in an optimal way. Then, an example is given for better understanding the framework. Particularly, besides good scalability, and failure recovery, it possesses other characteristics such as routing simplicity, self-organization, etc. In such an overlay mesh, an applicable source routing scheme called hierarchical source routing is used to transmit data packet based on UDP protocol. Finally, a guideline derived from a number of simulations is proposed on how to set various parameters in this overlay mesh, which makes the overlay more efficient.**

Index Terms: **Control law, hierarchical, overlay, source routing.**

I. INTRODUCTION

The initial design objective of Internet is to provide best effort (BE) transport service. However, along with the increase of Internet scale and users' demand, it is hoped that more and more functions and services can be applied, such as guarantee of quality of service, ensuring connection when link failure occurs. Researchers have studied these problems for years, but up to now, most of which have not been totally solved yet due to the following reasons. First, the hierarchical architecture of Internet is a tradeoff between efficiency and simplicity. Choosing proper location to implement new functions is not as simple as we supposed. An example is the failure to realize QoS in network layer. Second, original design goals of some architectures and protocols are not related even inconsistent to some new demands, such as multi-path and fast-failure-recovery (FFR). So "patching" the Internet with add-on service is sometimes not practically viable. A new system rather than tiny modification on original protocol may be the only approach to some special demands. Third, the variety of users' demands brings complexity of hardware and software to routers. Even if routers can tackle some issues in technical respect, the upgrade cost of routers in the whole Internet is too huge.

As a result, the concept of overlay networks has received a lot of attention recently. Its basic idea is to build an application layer network which lies on top of TCP/IP layer. Any path between two overlay nodes is seen as a virtual link, which corresponds to a default path between them. Since overlay is implemented at application layer, we just need to modify end hosts rather than routers to bring low deployment cost and easy maintenance. However, implementing new functions totally on user's host computer may be not fit for all applications. A special case is the framework put forward in this paper which still needs "overlay router" to accomplish the task of traditional routers. But there are essential differences between them in that the former does not work at IP layer but application layer. On the other hand, overlay inevitably brings inefficiency to transportation since its routing policy is usually metric-oriented which results in non-optimal distribution of traffic. Actually, a majority of systems including RON [1] and [2] lack a general scheme to deal with traffic-distribution problem. They are mainly applied to improve part of users' end-to-end performance, such as minimizing delay or maximizing throughput of a specified source/destination, not taking global performance into consideration. Besides, taking the limitation of node number in RON as example, the large amount of information of topology maintenance usually adversely affects scalability and efficiency. Therefore, we also apply an optimal algorithm to increase global efficiency, as well as hierarchical structure and routing scheme to deal with scalability.

We have the following contributions and results.

- A hierarchical overlay framework is proposed first. In order to enhance scalability, we adopt hierarchical structure where inner-area and inter-area can use any applicable link state protocol to organize its topology. Then a typical example is given with good scalability, routing simplicity and self- organization.
- A hierarchical source routing scheme which fits hierarchical topology and optimal algorithm is put forward. In our scheme, inner-area and inter-area transmit data packets through source routing, respectively. Each source routing node makes decision all by itself. By this way, each node just needs to maintain information of the area it belongs to, which diminishes its processing cost, and improves routing scalability as well. Moreover, this source routing scheme can be used by the optimal algorithm to allocate traffic among multiple paths from source to destination.
- Some insight is provided on how to set parameters in packet-based overlay mesh. A buffer management algorithm makes the control law work in real overlay mesh. Some surprising results derived from simulations display differences between overlay mesh using control law and traditional network.

The remainder of the paper is organized as follows: Section II reviews the end-to-end optimal traffic control law and related

Manuscript received May 15, 2007.

The authors are with the Department of Computer Science and Technology, at Tsinghua University, China. email: {liuchunyu, xuke}@csnet1.cs.tsinghua.edu.cn.

work. Section III presents the hierarchical overlay framework. Then a precise statement of implementation scheme of the control law in overlay mesh is presented in Section IV. In Section V, some insight is provided on how to design the parameters by simulations in order to address some of the problems that one encounters in a practical implementation. Finally, conclusions and directions for future research are given in Section VI.

II. THEORETICAL FOUNDATION AND RELATED WORK

In [3], a global optimal algorithm is proposed through which the traffic in a network can be allocated in the optimal mode. The algorithm is shown below that readers can have a rudimentary knowledge of it. To get more details, please refer to [3]–[5].

In a network, all traffic can be classified into some types of calls. Types here denote the aggregate of calls with the same ingress, egress, and class-of-service (CoS) requirement. Several CoS requirements are described which include the assured forwarding (AF), the minimum rate guaranteed service (MRGS), the upper bound rate service (UBRS), the minimum rate guaranteed and upper bound rate service (MRGUBR), and best effort (BE) service. The global optimizing object is

$$U(x) \doteq \sum_{i=1}^{n} U_i(x_i) \doteq \sum_{i=1}^{n} U_i(x_{i,1}, x_{i,2}, \cdots, x_{i,n_i})$$

where i is the ith type of call, n is the number of types of calls, and n_i is the number of path available for calls of type i.

The optimal resource allocation problem can be formulated as

$$\max_{x} U(x)$$

subjecting to the network capacity constraints, namely

$$\sum_{i,j:l\in P_{i,j}} x_{i,j} - c_l \le 0$$

where $P_{i,j}$ is the jth path of calls of type i, l denotes the physical link on it, and c_l is the capacity of link l.

The assured forwarding (AF) requirement:

$$\sum_{j=1}^{n_i} x_{i,j} = \Lambda_i, \quad i = 1, 2, \cdots, s_1.$$

The minimum rate guaranteed service (MRGS) requirements:

$$\sum_{j=1}^{n_i} x_{i,j} \ge \theta_i, \quad i = s_1 + 1, s_1 + 2, \cdots, s_2.$$

The upper bound rate service (UBRS) requirements:

$$\sum_{j=1}^{n_i} x_{i,j} \le \Theta_i, \quad i = s_2 + 1, s_2 + 2, \cdots, s_3.$$

The minimum rate guaranteed and upper bound rate service (MRGUBR) requirements:

$$\theta_i \le \sum_{j=1}^{n_i} x_{i,j} \le \Theta_i, \quad i = s_3 + 1, s_3 + 2, \cdots, s_4.$$

Notations Λ_i, θ_i, and Θ_i are bounds of traffic rate. Besides, all data rates are nonnegative

$$x_{i,j} \ge 0, \quad i = 1, 2, \cdots, n, \ j = 1, 2, \cdots, n_i.$$

Now, the novel family of distributed rate adaptation control laws is presented as follows.

Let $f_{i,j} \doteq (1 - e^{-\partial U/\partial x_{i,j}})$ and let $z_{i,j}(t,x)$ be positive scalar functions for all i and all j [5]. The family of control laws is as following.

For AF calls, let

$$\dot{x}_{i,j} = z_{i,j}(t,x)\big[f_{i,j}(x) - (1 - \overline{cg}_{i,j} r_i)\big]$$

where $r_i(x_i) = \begin{cases} r_{min} < 1, & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \Lambda_i \\ r_{max} > 1, & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \Lambda_i. \end{cases}$

For MRGS calls, let

$$\dot{x}_{i,j} = z_{i,j}(t,x)\big[f_{i,j}(x) - (1 - \overline{cg}_{i,j} r_i^m)\big]$$

where $r_i^m(x_i) = \begin{cases} 1, & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \theta_i \\ r_{max}^m > 1, & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \theta_i. \end{cases}$

For UBRS calls, let

$$\dot{x}_{i,j} = z_{i,j}(t,x)\big[f_{i,j}(x) - (1 - \overline{cg}_{i,j} r_i^M)\big]$$

where $r_i^M(x_i) = \begin{cases} r_{min}^M < 1, & \text{if } \sum_{j=1}^{n_i} x_{i,j} > \Theta_i \\ 1, & \text{if } \sum_{j=1}^{n_i} x_{i,j} < \Theta_i. \end{cases}$

For MRGUBS calls, let

$$\dot{x}_{i,j} = z_{i,j}(t,x)\big[f_{i,j}(x) - (1 - \overline{cg}_{i,j} r_i^m r_i^M)\big]$$

where r_i^m and r_i^M are defined as above.

For BE calls, let

$$\dot{x}_{i,j} = z_{i,j}(t,x)\big[f_{i,j}(x) - (1 - \overline{cg}_{i,j}\big].$$

Note that $x_{i,j}$ is derivative of current traffic rate in the jth path of call of type i, which reflects the trend and velocity of change of traffic in future. $cg_{i,j}$ is congestion feedback and it equals to 1 when the jth path of call of type i is congested, and 0, otherwise. And $\overline{cg}_{i,j} = 1 - cg_{i,j}$. Note that congestion feedback is the only information needed for end node from outside.

In [3], authors have proved that under some assumptions, the control law presented above converge to a traffic allocation that maximized the utility function $U(x)$ subjecting to capacity constraints of the network, CoS requirements and non-negativity of all the data rates.

However, this family of control law is just an ideal case, where flow and iteration steps are both fine-grained, and congestion feedback is global information which can be derived by end node almost immediately. In real network environment, flows consist of packets, whose size can not be neglected.

In addition, the magnitude of delay from congestion occurrence to congestion detection by end host is second. Thus, how the control law works in real network is worth studying further. In addition, even if the control law works well when some ideal assumptions are relaxed, how to deploy it in real network also

deserves special attention. Source routing is one of the requirements, while almost all routers do not support source routing. So a new framework is needed to bring this theory into use.

There is a lot of literature on distributed traffic control protocols that can quickly react to network congestions and link/node failures. Various solutions have been developed at different layers independently, such as distributed, multi-path enabled traffic engineering (TE) and FFR at the networking layer (e.g., TeXCP [6] and Mate [7]) and overlay (e.g., RON [1] and OverQoS [8]), and protocols at the transport layer with various degrees of involvement of network nodes (e.g., Fast TCP [9], VCP [10], and XCP [11]). They are available in some cases, but they confront various problems like scalability and cost, etc.

Overlay is a good solution to deploy new services that cannot practically be embedded directly in traditional Internet, such as multicast [12], [13], file sharing [14], [15], QoS overlay [8]. Pure overlay rather than end-system overlay was studied by some prototypes or frameworks, for example, RON [1], SON [2], Spines [16], [17], OverQoS [8], [18], etc. On the other hand, many P2P applications, such as Gnutella [19], Coolstreaming [13] can be regard as overlay networks.

However, scalability is a main problem of some existing overlays. Luckily, hierarchical overlay structure can well tackle this issue. The algorithm proposed in [20] is particularly relevant to the work presented in this paper. It introduced some key steps in construction and maintenance processes of hierarchical P2P network based on Yao-Graph. Our model has been influenced by it in that there are also super nodes to divide the whole mesh into small areas for high extensibility. But in our model, a stable control center out of the mesh can better simplify construction and maintenance of the mesh.

In addition, Jiang *et al.* [21] proposed a zone approach of overlay multicast that performs hierarchical routing on two-levels. A hierarchical P2P system for distributed software development and lookup service were put forward by Bischofs [22] and Garces-Erice [23], respectively. As we can see, hierarchical structure is a generic approach to deal with problem of scalability.

Recently, sliding mode control theory is used by a series of literature [3]–[5], [24], [25] to solve distributed traffic control problem, whose primary contribution is that their theory which is called optimization-based distributed control laws (ODCLs) can overcome most of the limitations suffered by the existing distributed traffic control protocol, allowing distributed, mutipath, multi-CoS load balancing, and making the traffic distribution optimized which is measured by a given utility function. However, they just give the rules and calculate results in a nearly continuous-time, fluid-flow model.

Based on studies in literatures above, Che *et al.* [26] proposed a new structure for future Internet where the control law is involved at both transport layer and networking layer. To integrate end-to-end and edge-to-edge traffic control structure together, a multi-layer, multi-domain overlay network can achieve rich QoS features, TE, and FFR in theory. This paper further proves the feasibility of overlay network when using control law and provides its implementation scheme. Our study differs from this reference in that we just apply the control law in application layer without modifying underlying protocols. However, it achieves the same features above in packet-based network environment.

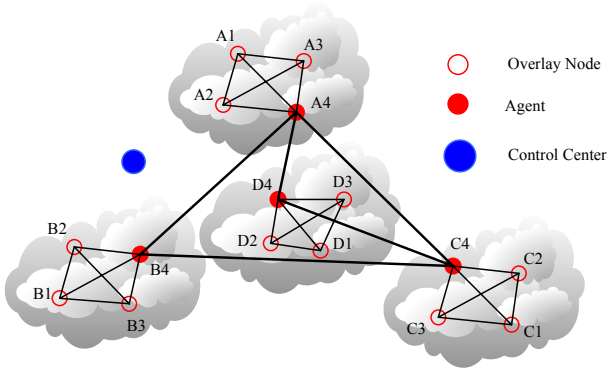


Fig. 1. Overlay mesh.

III. HIERARCHICAL STRUCTURE FRAMEWORK OF OVERLAY MESH

For purpose of scalability, this paper adopts a hierarchical structure to form overlay mesh topology. The main design idea can be described as.

Through any given neighbor detection protocol (one of which is depicted in following section), each overlay node can connect with other nodes to construct a small-scale overlay sub-mesh (area) whose topology can be formed by any link state protocol. Each area has some best-performed overlay nodes to be its agents (or called super nodes), which take charge of registering to the control center of the whole overlay mesh and communicating with other areas, and all agents form an upper hierarchical mesh, or called agent-mesh, through any link state protocol. In this way, the whole overlay mesh forms two layers, the higher one of which is constructed through centralized registration, and each agent is like edge router of its area, while the lower one of which is formed by self-organized method to get topology information in a small range, and these overlay nodes act as inner-area routers.

A special hierarchical network is shown in Fig. 1. Each area only has one agent and each node has a bounded in-degree and out-degree. So the connections in both intradomain and interdomain may not be complete graph. Each normal node just knows information about its area by link state protocol, and information outside the area is not advertised to inside. Agent knows information about its area and agent-mesh. We will choose this overlay mesh model to deploy the control law for simplicity.

In Fig. 1, each link between two overlay nodes corresponds to a default Internet path. It can be seen as a virtual link from application layer's point of view. In order to select better paths for transmitting data groups, overlay nodes can measure some metrics of each virtual link, including available bandwidth, delay, loss rate, and jitter, etc.

In the worst case, topologies in area and agent-mesh are both complete graph. Assumed that each area has average M nodes, common overlay node just needs to maintain $M-1$ virtual links, and the number of agent's virtual links is only $M+N/M-2$, where N is the number of nodes of the graph. When M is $O(\sqrt{N})$, virtual link number of each overlay node

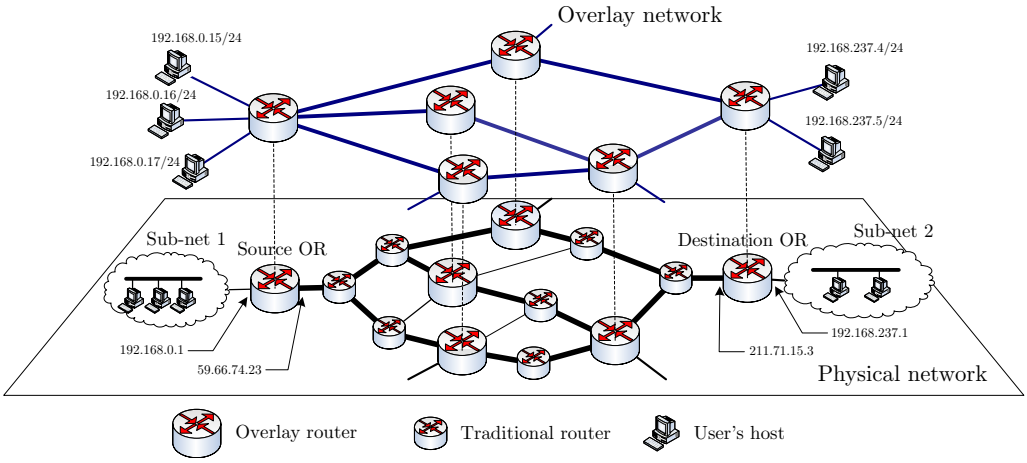


Fig. 2. Overlay mesh and traditional Internet.

is also $O(\sqrt{N})$, which can be regarded as good scalability. The case can become better if other topologies instead of complete graph are adopted.

Furthermore, link state protocol itself can provide fast failure recovery. If we use hello message to maintain neighbor relationship, virtual link failure or node collapse can be detected after not receiving hello message for several times. If active measurement is taken, this process will be shorter. Then virtual link state advertisement (VLSA) flooding can inform other nodes about the failure, followed by choosing new path by source node.

IV. IMPLEMENTATION SCHEME OF CONTROL LAW

A. Hierarchical Overlay Mesh Organization and Maintenance

In this section, we introduce the implementation scheme of end-to-end optimal traffic control law in hierarchical overlay mesh. To implement in overlay rather than traditional network results from a large number of differences between traditional routing policy and source routing used in the control law.

Fig. 2 illustrates how the overlay is organized. Overlay nodes act as routers in overlay mesh to transmit user's data group in application layer, so they are called overlay routers also. For clarity, only connections rather than hierarchical structure are shown. Each overlay node is the access point of its local sub-net (such as Ethernet or Token-Ring), and it can play the role of NAT (Network Address Translation). Actually, implementing the control law in a NAT gateway is a good approach to satisfy CoS as well as to reuse finite address space. The overlay mesh works according to the guidelines below:

- *Registration phase*: A new overlay node joins the mesh. It firstly sends HELLO message to control center to register. The feedback message from control center is attached with addresses of all super nodes. If being the first node in mesh, it becomes super node.
- *Choose phase*: The new node measures QoS metrics of the virtual link between each super node and itself. It chooses the best node and sends JOIN message to it. Return message contains addresses of all members in this area. If metrics to all super nodes are below a threshold, the node itself will become super node and inform the control center.
- *Join phase*: The new node randomly chooses n neighbors in all members and sends messages to them. This process does not finish until n neighbors return agreement messages or all members have been tried. First 3 phases can be seen in Fig. 3.
- *Adjustment phase*: All nodes in an area periodically select a best-performed node to be their new agent to replace the old one. However, we do not hope this happens frequently, so we need to select a good-performed, stable, trusty, and secure agent, which is out of scope of this study.
- *Topology maintenance phase*: When a node joins or leaves, it should send VLSA to its neighbors, which includes virtual links and reachable sub-net information taken charge by itself. Each node which receives the VLSA then forwards it to neighbors except the one the VLSA comes from. Super node clusters all sub-net information in its area and broadcasts it to other super nodes which in turn broadcast this message into its own area.
- *Data Transmit phase*: Each overlay node has a list of reachable address. When user's group arrives, overlay node firstly looks up its destination address in the list. If the address matches an item and its sub-net mask, this group and a header will be encapsulated by UDP protocol, and sent to next hop. Otherwise, the source address of the group will be translated by NAT protocol, and forwarded to its destination through traditional Internet.

With regard to topology maintenance, we should distinguish two main cases according to leaving node:

- *Normal node*: When a neighbor detects its leaving by VLSA or active detection, it will retransmit VLSA to neighbors except the one VLSA received from. After flooding, all nodes in this area get the leaving information and are able to update their topology database.
- *Super node*: After detecting its leaving, a piece of VLSA is also flooded in the area and agent-mesh. Then another best-performed agent in this area will be selected and contact with control center to reconnect to other agents.

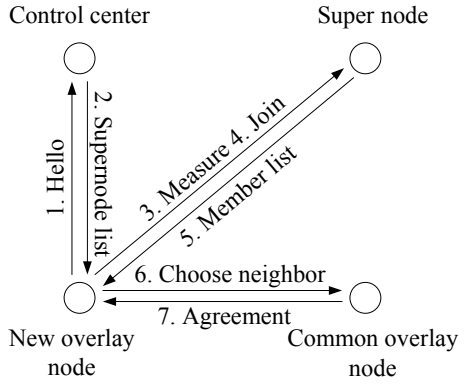


Fig. 3. Steps of new overlay node's join.

B. Source Routing Scheme for Control Law

The source routing is hard to use in practice for reason of scalability. In our overlay mesh, however, it is a necessary part for multi-path transmission. In this section, we put forward a feasible source routing scheme especially in hierarchical overlay mesh for data forwarding.

Through topology organization and maintenance scheme outlined above, all nodes in an area hold the same topology information by which each common node can choose first n shortest paths to any other node in its area. When user's group arrives, if its destination belongs to another overlay node (egress) in the same area, it is transmitted to its destination along the first n shortest paths. Because of small amount of nodes in an area, source routing is feasible. The format of packet header used by source routing is shown in Fig. 4.

- # of path: Amount of paths to transmit user's packets
- Path no.: The number of path
- Length: Length of header, unit is 32-bit
- Option: Unused
- Destination address: Destination IP address of packet
- Source address: Source IP address of packet
- First hop address: IP address of the first overlay node to which the ingress decides to send the packet.
- Second hop Address: Next hop to which the first hop node will send the packet. Following fields are the 3rd, the 4th,$\cdots$, the last hop.

Take the mesh in Fig. 1 for example, assumed A1 sends packets to A3, the headers are shown in Figs. 5(a) and 5(b). As we can see, there are overall two paths, one of which is a direct path and the other passes a relay node A2.

If destination of the group belongs to another area, for example, A1 sends packets to C1, the ingress node A1 firstly sends the packet to its agent A4, and the packet headers are shown in Figs. 5(c) and 5(d). It needs special attention that the path actually corresponds to two paths from point of view of ingress node A1 due to different *Path no*.

When agent A4 receives the first packet of this flow, and the destination is not itself, it will check the field *# of path*. Since it holds all topology information of agent-mesh, it can choose first two shortest paths to C1's agent by source routing. For example, it chooses A4-C4 and A4-D4-C4. The packet headers are shown in Figs. 5(e) and 5(f), from which we can see that the agent A4 modifies routing fields according to its source routing decision.

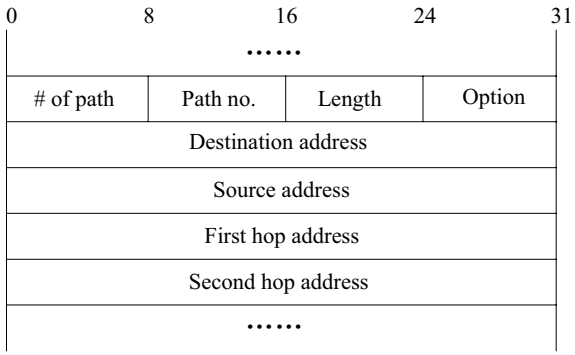


Fig. 4. Packet header format.

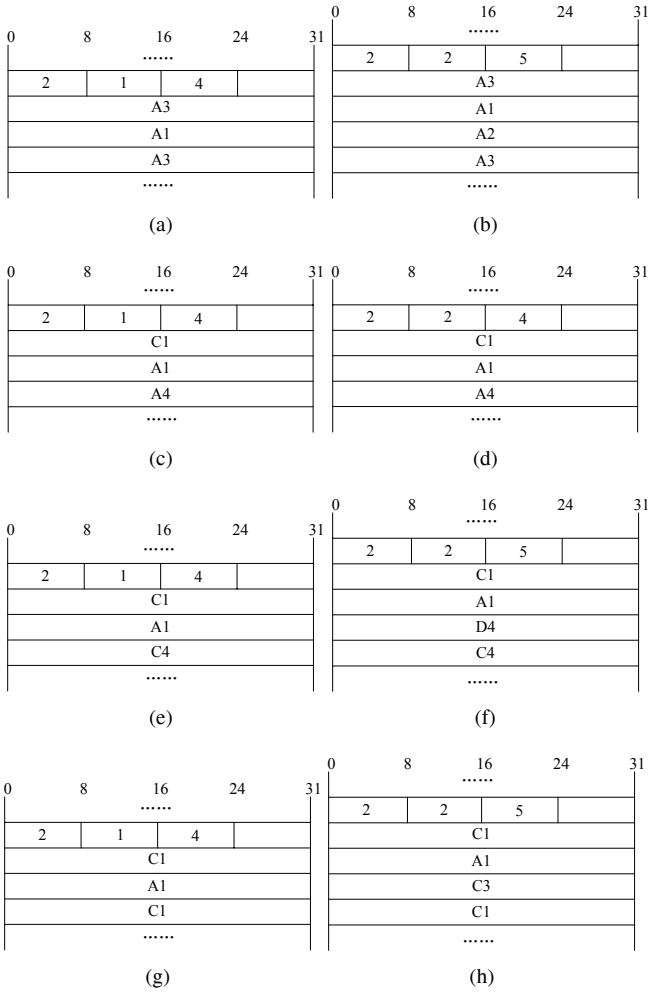


Fig. 5. Packet header examples.

The same things are done by agent C4, and the headers change to the case in Figs. 5(g) and 5(h).

From source routing scheme above, we can see that for multiple-area transmission, this scheme changes traditional source routing into three phases, which are routing in source area, agent-mesh and destination area, respectively, and each source routing node can make decision all by itself. It resolves the problem of scalability of source routing and makes the control law feasible in real network.

V. SIMULATION EXAMPLES AND ANALYSIS

In this section, simulation examples are presented, which help us make a clear understanding on how the control law works in packet-based overlay mesh rather than in a continuous-time, fluid-flow model.

A. Simulation Setup

We concentrate on the case where both BE and AF traffic share the overlay mesh since one would obtain similar results if one would also have MRGS, UBRS, and MRGUBS Cos present [5].

In order to make the simulation more trustworthy, we used real topology where 15 overlay nodes distribute in 15 core nodes in CERNET (China Education and Research Network) to conduct our simulations. The connections and distribution of overlay nodes are shown in Fig. 6(a) and its logical topology can be summarized as Fig. 6(b). All nodes in our model are overlay nodes, and links are virtual links which correspond to the default Internet path between nodes. Thus, the capacity of each virtual link is actually available bandwidth and fluctuates all the time. It should be noticed that the control law requires that link capacities are all constant, since it tries to keep all traffic at the critical point of congestion. Time-varying capacities inevitably result in the failure of this behavior. However, in computation process of the control law, which is operated by literatures related, congestion completely depends on incoming and outgoing aggregated data rate at node, and congestion occurs when incoming rate is higher than outgoing rate at node. But in real overlay routers, congestion happens just at the moment buffer is full, and it is not related to incoming and outgoing data rate directly. Thus, we can still employ the control law in the case with time-varying link capacities. Through buffer management algorithm below, variation of available bandwidth can be smoothed and its impact is greatly reduced so that the control law can work well.

To prove our viewpoint, the control law was tested in cases with constant and time-varying available bandwidth. Throughout the simulation, the neighbor relationship of nodes does not change, while the available bandwidth is set as $c = originalValue \cdot (0.3\sin(0.021t) + 1)$ in time-varying case, which simulates long-period variation of available bandwidth in real network. In order to show the impact of rapid variation, the available bandwidth also fluctuates in the range of $\pm 20\%$ with respect to c. Due to limitation of simulation, the bandwidth can not vary continuously. However, in simulation which is discrete in nature, a shorter discretization step can better simulate continuous varying of bandwidth in real world. We set the change interval as 100 ms, namely the capacity of each virtual link changes 10 times per second, and the bandwidth can fluctuate over the interval $[0.1c, 6.2c]$ in a second. One can use shorter interval, such as 10 ms, but we believe that it will not impact the result for existence of network latency and buffer management algorithm.

There are overall $n = 8$ types of calls and the paths for each one are indicated in Table 1, where n_i is the number of paths available for calls of type i. The utilization function is shown below

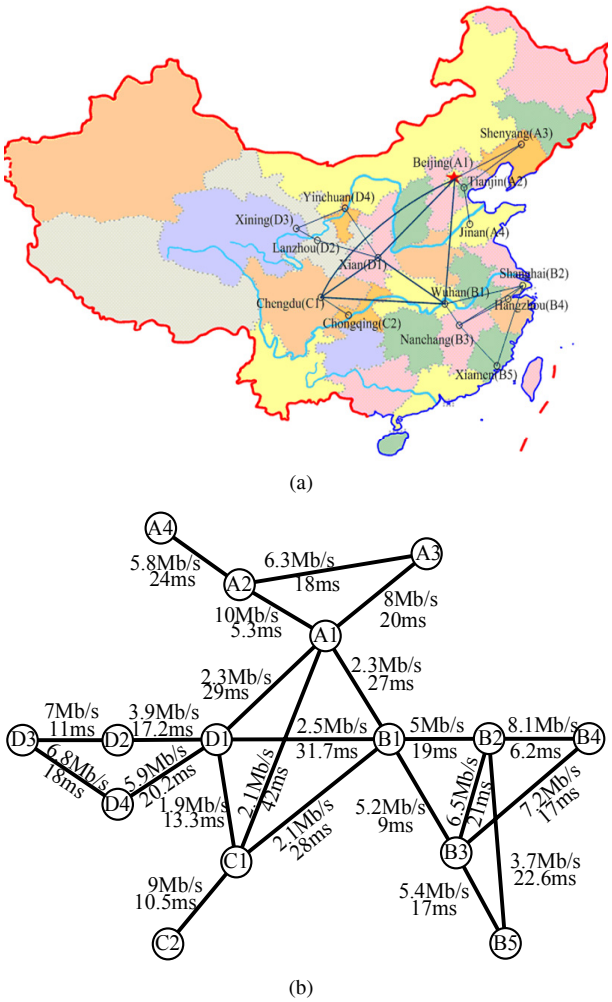


Fig. 6. Simulation topology: (a) 15 overlay nodes distributed in CERNET, China, and (b) topology of the overlay mesh.

$$U(x) = \sum_{i=1}^{8} 0.1 \log\left(0.5 + \sum_{j=1}^{n_i} x_{i,j}\right).$$

We used logarithm function since it is a typical concave function which increases rapidly at the beginning and slows down afterwards. It can avoid the case that links are dominated by a single flow, since utility value can be higher when the link is shared by a number of flows. The AIMD mechanism used in TCP is also for the same reason. The term 0.5 in formula is to avoid an infinite derivative at 0 traffic rate, and the term 0.1 can make the utility value in a proper range and does not affect the result.

As for the AF service requirement, calls of type $i = 3,\ 5$ are guaranteed to have rates 3 Mb/s and 2 Mb/s, respectively.

Given this, the control laws are of the following form:

For $i = 3,\ 5$ and $j = 1,\ 2$, i.e., AF calls, we have

$$\dot{x}_{i,j} = z_{i,j}\left[\left(1 - e^{-0.1\left(\sum_{j=1}^{n_i} x_{i,j} + 0.5\right)^{-1}}\right) - (1 - \overline{cg}_{i,j} r_i)\right]$$

where

$$r_i(x_i) = \begin{cases} 0.5, & \text{if } \sum_{j=1}^{n_i} x_{i,j} > 1 \\ 2, & \text{if } \sum_{j=1}^{n_i} x_{i,j} < 1. \end{cases}$$

Table 1. Paths available for each type of calls.

Type 1 - n_1=4	Type 2 - n_2=3	Type 3 - n_3=2	Type 4 - n_4=3
$x_{1,1} : A_3A_1B_1B_3B_5$ $x_{1,2} : A_3A_1B_1B_2B_3B_5$ $x_{1,3} : A_3A_1D_1B_1B_2B_5$ $x_{1,4} : A_3A_1C_1B_1B_2B_5$	$x_{2,1} : A_3A_1B_1C_1C_2$ $x_{2,2} : A_3A_1D_1C_1C_2$ $x_{2,3} : A_3A_1C_1C_2$	$x_{3,1} : A_4A_2A_1B_1B_3B_5$ $x_{3,2} : A_4A_2A_1D_1B_1B_3B_5$	$x_{4,1} : C_2C_1A_1A_2A_4$ $x_{4,2} : C_2C_1D_1A_1A_2A_4$ $x_{4,3} : C_2C_1B_1A_1A_2A_4$
Type 5 - n_5=2	Type 6 - n_6=4	Type 7 - n_7=2	Type 8 - n_8=3
$x_{5,1} : B_4B_2B_1D_1D_2D_3$ $x_{5,2} : B_4B_2B_3B_1C_1D_1D_2D_3$	$x_{6,1} : D_3D_2D_1B_1B_2B_4$ $x_{6,2} : D_3D_4D_1B_1B_2B_4$ $x_{6,3} : D_3D_4D_1C_1B_1B_3B_4$ $x_{6,4} : D_3D_2D_1C_1B_1B_3B_4$	$x_{7,1} : A_4A_2A_1A_3$ $x_{7,2} : A_4A_2A_3$	$x_{8,1} : B_4B_2B_5$ $x_{8,2} : B_4B_3B_5$ $x_{8,3} : B_4B_2B_3B_5$

Table 2. Default parameter settings.

Parameters	**Default values**
Assured traffic	$x_3 = 3$ Mb/s, $x_5 = 2$ Mb/s
Discretization step	100 ms
Calculation step	100 ms
Router buffer size	2 s
Data packet size	1000 bytes
Congestion feedback size	50 bytes

And for $i = 1, 2, 4, 6$ and $j = 1, \cdots, n_i$, i.e., BE calls

$$\dot{x}_{i,j} = z_{i,j}\left[\left(1 - e^{-0.1\left(\sum_{j=1}^{n_i} x_{i,j}+0.5\right)^{-1}}\right) - (1 - \overline{cg}_{i,j})\right]$$

where $z_{i,j}$ function for oscillation reduction is taken as $z_{i,j}(t) = \omega(t - t_0)$, where $\omega(t) = 0.4(0.25 + 0.5^t)$.

Finally, the continuous control law must be discretized and the discretization step should be big enough since packet size and calculation cost can not be neglected, and transmission can be finished in a discretization step. Then the discrete-time counterpart of control law is

$$x^d_{i,j}[(k+1)t_d] = x^d_{i,j}[kt_d] + t_d\dot{x}(kt_d), \quad k = 0, 1, \cdots$$

where t_d is called *calculation step* and its value in our packet-based simulation is 100 ms by default.

Furthermore, we assume that router's buffer size is proportional to its transmission performance, namely egress bandwidth. On the other hand, we wish to get a general result rather than the one which fits nothing but this topology and configuration. So we use unit *second* instead of MB or kB to denote buffer size. Anyone can compute the buffer size by multiplying real bandwidth in a specified network by the recommended value derived from our simulations. At the very beginning, we set 2 s as its default value which means sending out all packets costs the router 2 s when buffer is full. Besides, data packet size is 1000 bytes and congestion feedback is 50 bytes by default. In addition, we use a random sending rule for comparison, whose packet size and paths are the same as above. In this case, however, packets will be sent to paths available with the same probability. The rates will be increased slowly to obtain the maximum utility value.

In traditional network, packet loss indicates congestion happening somewhere along the path. Nevertheless, because of lacking AIMD (additive increase multiplicative decrease) mechanism which is used in TCP, a large number of packets will be dropped once congestion occurs in our overlay mesh. To avoid this phenomenon, router should send congestion information to source before congestion happens. Thus, an improved algorithm is designed to deal with this problem, in which a threshold is set to divide the buffer into two parts. The router will send congestion feedback once the size of buffer in use exceeds the threshold, and a packet is dropped only when the buffer is full. In addition, congestion feedback caused by a single packet should be sent only once at most by different nodes along path. A bit in *option* field of packet header is enough to distinguish whether congestion feedback has been sent or not. If not, the node will send this feedback to source, and set the bit as 1. Henceforth, the congestion feedback for this packet will not be sent any more at following nodes. As mentioned above, this algorithm can also help the control law works effectively in the case with time-varying available bandwidth, since it changes the control law from rate adaptation at critical point of congestion to buffer management around the threshold. By this algorithm, node has enough time to inform the source of flow about accelerating or decelerating the sending rate to respond to the variation of available bandwidth as long as the buffer is big enough. The algorithm is illustrated in Table 3.

Table 3. Buffer management and congestion feedback sending algorithm.

```
1   threshold ← default value
2   bufferInUse ← 0
3   void reveive( packet )
4   if bufferInUse + packet.size > bufferSize
5       drop packet
6       return
7   else
8       buffer ← buffer + packet
9       bufferInUse ← bufferInUse + packet.size
10  endif
11  if bufferInUse > buffer × threshold &&
    packet.congestBit == 0
12      send congestion feedback
13      packet.congestBit ← 1
14  endif
```

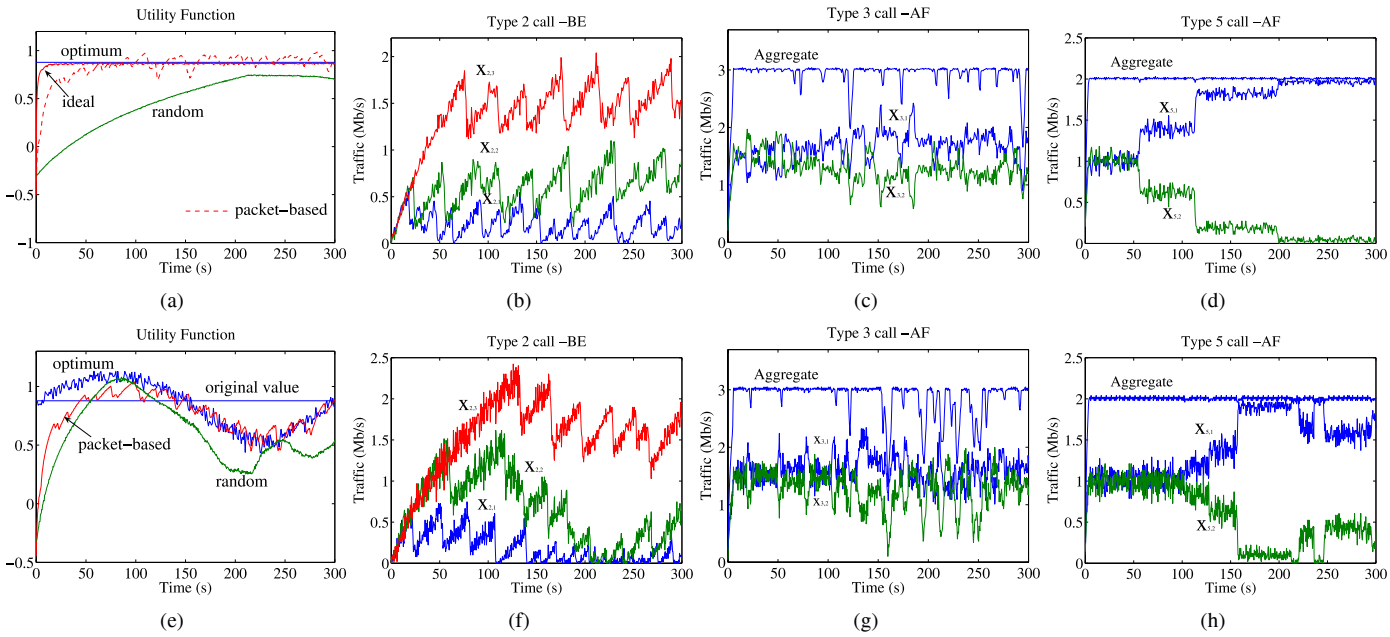


Fig. 7. Utility value and traffic of different types of calls.

B. Default Case

We compared utility value and traffic rates in cases with constant and time-varying available bandwidth, respectively. Due to space limitations, only calls of type 2, 3, and 5 are shown. In Fig. 7, the upper four are the constant case, and the lower four are time-varying one. Firstly, we observe that whatever available bandwidth is constant or not, the utility function based on real traffic rate always converges to its optimum value, which shows our buffer management algorithm does work well, and make the control law available in real network. In Fig. 7(a), the ideal curve is calculated by MATLAB, whose packet size is small enough to be allocated among paths almost continuously. The optimum curve in Fig. 7(e) corresponds to theoretical maximum utility value on time-varying available bandwidth condition and the original value corresponds to optimum value in Fig. 7(a) for a clear comparison. As what we expected, utility function value of using control law base on packet in overlay mesh is better than that of random sending in both cases. Actually, when taking logarithm operation into consideration, the performance of control law is much better than that of randomly sending. But the phenomenon that the utility value of using control law is partly larger than that of ideal case even the optimum one is a little surprising. This is due to that buffer can take in more traffic than maximum temporarily. Moreover, in Figs. 7(a) and 7(b), we can see that crests and hollows of curves alternate, since the fluctuation around buffer threshold will accelerate and decelerate source sending rate alternately. The rate adaptation caused by fluctuation of available bandwidth can be seen in Fig. 7(e). It mainly owes to buffering and buffer management algorithm. These results prove that the control law is worthy of deployment in overlay mesh from the point of view that it can make the most of bandwidth resource of network. Besides, the AF requirements imposed on calls of types $n = 3$ and $n = 5$ are guaranteed on the whole.

C. Calculation Step

We choose different calculation steps to measure their influences. Utility function as well as available throughput which just takes payload into account is considered. In Fig. 8, besides four utility curves converging to the same optimum value, we also find that the utility value is a little more fluctuant when calculation step becomes longer. Moreover, the available throughput of the cases using longer calculation step does not reduce too much (Fig. 9), and they are all higher than that of random sending case. However, longer calculation step makes the source be more insensitive to congestions which results in higher loss rate, which is illustrated in Table 4. Thus, the calculation step should be as small as possible.

D. Packet Size

Packet size is a parameter which has impact on the effect of using control law. In simulation, the packet size was set as from 300 bytes to 1500 bytes, and we assume that packet header size is 50 bytes. On the one hand, intuitively the larger the packet is, the relatively smaller the overhead of header will be. Fig. 10 also shows the available throughput of each case which demonstrates that larger packet size brings higher throughput to the whole mesh. On the other hand, however, if packet is excessively large, the control granularity becomes so large that it may result in that the traffic rate can not increase or decrease continuously, which may affect the validity of control law. In real network environment, the user's packet size is usually limited, and is less than 1500 bytes especially in Ethernet. Besides, MTU of router is usually set as about 1500 bytes. Although overlay node can use large UDP packet in order to decrease the overhead of packet header, routers will perform fragmentation, breaking the packet up into smaller pieces which are smaller than MTU, and it will reduce the transport efficiency. Thus, it is recommended that packet size is about 1500 bytes.

Table 4. Packet loss rate of different calculation steps.

Calculation step (ms)	50	100	200	300	400	500	600	700	800	900	1000
Loss Rate (0.1%)	0.085	0.075	0.053	0.204	0.153	0.332	0.808	1.647	0.545	1.567	1.827

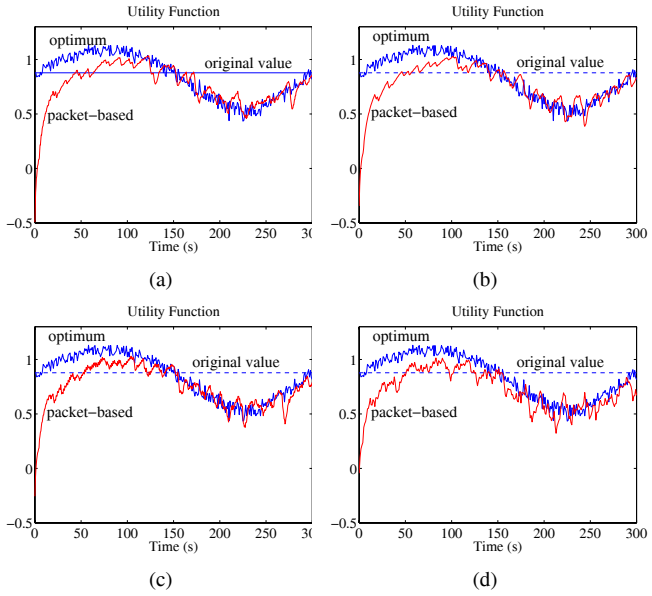


Fig. 8. Utility value for different calculation steps: (a) Calculation step=50 ms, (b) calculation step=200 ms, (c) calculation step=500 ms, and (d) calculation step=1000 ms.

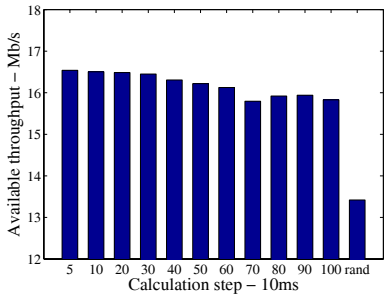


Fig. 9. Available throughput for different calculation steps.

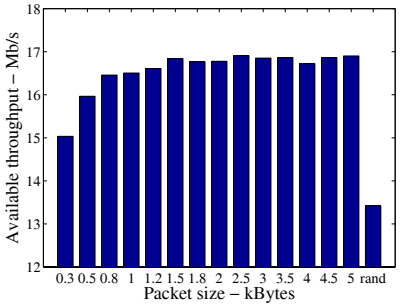


Fig. 10. Available throughput for different packet sizes and random sending.

E. Buffer Size

Another important parameter that has a bearing on the performance of the control law is the buffer size of router. We will compare the value of utility function in some scenarios where buffer sizes and threshold are set as different values. In Figs. 11(a) and 11(b), just as what we expected, excessively large threshold results in relatively high packet loss rate, and excessively small threshold brings low throughput. But to our surprise, keeping the threshold fixed, the larger the buffer size is, the higher the loss rate and the lower the throughput will be. When buffer size is relatively large, the source node is prone to send more packets than maximum, since they can be buffered by routers, and congestion does not happen for a while. Once congestion occurs, the extra packets will induce a great deal of congestion feedback, which will result in rate decreasing to a lower level in the source node afterwards. When the buffer in use slowly declines below the threshold, the rate will increase once again. The larger the buffer, the longer the period of this phenomenon will be, and the more violently traffic rate will fluctuate as well. This effect can be partly confirmed by the wide-range fluctuation of utility function curves shown in Fig. 12. So, moderate buffer size and threshold can maximize the available throughput as well as minimizing packet loss rate. In practical implementation, the buffer size which can support sending for 4 seconds is enough for each overlay node, and the threshold can be set as 40%.

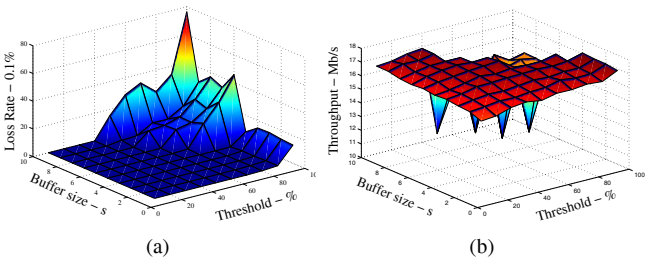


Fig. 11. In different buffer sizes and thresholds: (a) Lossrate and (b) throughput.

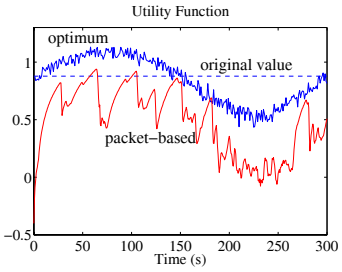


Fig. 12. Utility function when buffer size=5s, threshold=80%.

The control law has a salient feature: Robustness with respect to link and node failures. Just as what we mentioned above, however, overlay mesh itself is able to detect failures by any link state protocol. Source node can make a rapid response to failures through switching the flow from failure path to another. So our overlay mesh can also achieve FFR.

VI. CONCLUSION

In this paper, we introduced a hierarchical overlay framework in which the end-to-end optimal traffic control law is embedded. It integrates nice features in both overlay and control law, including CoS, traffic engineering, and FFR. Then an implementation scheme including mesh organization and routing protocol was brought forward. According to our framework and scheme, a scalable overlay mesh can be designed and implemented in present Internet. Moreover, a buffer management algorithm and congestion feedback sending algorithm were also put forward to deal with the problems in our framework. Simulation studies showed that how to set parameters in practical implementation. A short calculation step, with 1500-byte packet size, moderate buffer size and threshold are preferred. Some choices are a bit surprising but can really make the control law work better.

As mentioned above, the main focus of this paper is the introduction about framework and implementation scheme. Therefore, a possible direction for future research is to test the implementation in large-scale network settings, and to develop a prototype based on the control law. It is believed that to deploy a practical system which makes the most of limited network resource is quite attractive.

VII. ACKNOWLEDGMENTS

This work has been supported by NSFC (No. 60473082), 973 Project (No. 2003CB314801), and 863 Project (No. 2006AA01Z209).

REFERENCES

[1] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay network," in *Proc. ACM SOSP*, Oct. 2001, pp. 131–145.

[2] Z. Duan, Z.-L. Zhang, and Y. T. Hou, "Service overlay networks: SLAs, QoS, and bandwidth provisioning," in *Proc. IEEE Int'l Conf. Network Protocols*, 2002.

[3] B. A. Movsichoff, C. M. Lagoa, and H. Che, "End-to-end optimal algorithms for integrated QoS, traffic engineering, and failure recovery," to appear in *IEEE/ACM Trans. Netw.*, Nov. 2007.

[4] C. Lagoa and H. Che, "Decentralized optimal traffic engineering for the Internet," *ACM SIGCOMM Computer Commun. Rev.*, vol. 3, no. 5, Oct. 2000.

[5] C. Lagoa, H. Che, and B. Movsichoff, "Adaptive control algorithms for decentralized optimal traffic engineering," *IEEE/ACM Trans. Netw.*, vol. 12, no. 3, June 2004.

[6] S. Kandula, D. Katabi, B. Davie, and A. Charney, "TeXCP: Responsive yet stable traffic engineering," in *Proc. ACM SIGCOMM*, 2005.

[7] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," in *Proc. IEEE INFOCOM*, 2001.

[8] L. Subramanian, I. Stoica, H. Balakrishnan, and R. H. Katz, "OverQoS: Offering Internet QoS using overlays," in *Proc. HotNet-I Workshop*, Oct. 2002.

[9] C. Jin, D. Wei, and S. Low, "FAST TCP: Motivation, architecture, algorithms, and performance," in *Proc. IEEE INFOCOM*, Mar. 2004.

[10] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One more bit is enough," in *Proc. ACM SIGCOMM*, 2005.

[11] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," in *Proc. ACM SIGCOMM*, 2002.

[12] Y. H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *Proc. ACM SIGCOMM*, June 2000, pp. 1–12.

[13] X. Zhang, J. Liu, B. Li, and T. S. P. Yum, "Coolstreaming/DONet: A data-driven overlay network for efficent media streaming," in *Proc. IEEE INFOCOM*, 2005.

[14] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area cooperative storage with CFS," in *Proc. the 18th ACM Symp. Operating Systems Principles (SOSP)*, Alberta, Canada, Oct. 2001.

[15] M. Ripeanu, I. Foster, and A. Iamnitchi, "Mapping the Gnutella network: Properties of large-scale peer-to-peer systems and implications for system design," *IEEE Internet Comput. J.*, vol. 6, no. 1, 2002.

[16] C. Danilov, "Performance and functionality in overlay networks," A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy, Baltimore, Maryland, Sept. 2004.

[17] Spines, [Online] Available:http://www.spines.org/

[18] L. Subramanian, I. Stoica, H. Balakrishnan, R. Katz, "Over QoS: An overlay based architecture for enhancing Internet QoS," in *Proc. 1st Symp. Networked Systems Design and Implementation*, San Francisco, CA, Mar. 2004.

[19] GNUTELLA.WEGO.COM. Gnutella: Distributed information sharing,2000. [Online] Available:http://gnutella.wego.com/

[20] M. Kleis, E. Lua and X. Zhou, "Hierarchical peer-to-peer networks using lightweight superpeer topologies," in *Proc. 10th IEEE Symp. Computers and Commun.*, Cartagena, Spain, June 2005.

[21] Y. Jiang, M. Wu and W. Shu, "A hierarchical overlay multicast network," in *Proc. IEEE Int'l Conf. Multimedia and Expo*, 2004.

[22] L. Bischofs and W. Hasselbring, "A hierarchical super peer network for distributed software development," in *Proc. Workshop on Cooperative Support for Distributed Software Engineering Processes*, 2004.

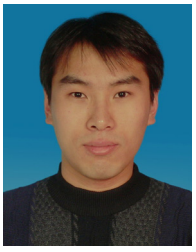
[23] L. Garces-Erice, E.W. Biersack, P.A. Felber, K.W. Ross, and G. Urvoy-Keller, "Hierarchical peer-to-peer systems," in *Proc. ACM/IFIP Int'l Conf. Parallel and Distributed Comput.* 2003.

[24] B. Movsichoff, C. Lagoa, and H. Che, "Minimal feedback optimal algorithms for traffic engineering in computer networks," accepted for publication in *IEEE/ACM Trans. Netw.*, [Online] Availabe: http://crystal.uta.edu/ hche/ PUBLICATIONS/publication.htm

[25] B. Movsichoff, C. Lagoa, and H. Che, "Decentralized optimal traffic engineering in connectionless networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 2, pp. 293–303, Feb. 2005.

[26] H. Che, W. Su, C. Lagoa, K. Xu, C. Liu, and Y. Cui, "An integrated, distributed traffic control strategy for the future Internet," in *Proc. SIGCOMM*, Pisa, Italy, Sept. 11–15, 2006.

[27] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, "Search and replication in unstructured peer-to-peer networks," in *Proc. Int'l Conf. Supercomputing*, 2002



Chunyu Liu was born in China. He received his B.E. degree from Renmin University of China in 2005. Currently, he is an MPhil candidate in the Department of Computer Science and Technology at Tsinghua University, China. His research interests include P2P/overlay network, traffic engineering, and game theory. His personal hobbies include sports and reading.

Ke Xu received the B.S., M.S., and Ph.D. degrees in computer science from Tsinghua University, China in 1996, 1998 and 2001, respectively. Currently he is an Associate Professor in the department of computer science of Tsinghua University. His research interests include next generation Internet, switch and router architecture, P2P, and overlay network. He is a member of IEEE and IEEE Communication Society.