文章编号：1008-7826（2006）01-0027-04

基于隧道 agent 的端到端 QoS

许华荣 [1,2]

(1.福建师大福清分校 数学与计算机系, 福建 福清 350300; 2.厦门大学 物理系, 福建 厦门 361005)

摘　要: 目前，在 IP 网上数据的通信可通过 RSVP 实现端到端的 QoS. 但在数据流传送的路径上有不支持 RSVP 的路由器，就不能利用 RSVP 实现端到端的 QoS. 本文提出把不支持 RSVP 的路由器组成隧道，并利用隧道 agent 实现端到端的 QoS.

关键词: 隧道 ; agent ; QoS ; RSVP

中图分类号:　TP393.1　**文献标识码:**　A

1　引言

目前，IP 网已获得广泛的应用，但早期的 IP 部署忽略分组报头的 ToS(Type of Service)字节，路由器也不使用它来影响 IP 分组的转发策略. 这样 Internet 就未能支持 QoS，只提供所谓尽力而为（Best-effort）级别的服务. 其主要思想是当用户提出服务请求时，网络不会拒绝用户的请求，但是服务的质量要依据当时网络的状况而定，分组的丢失率、出错率和传输延迟是不能确定的.

为了使 IP 网提供 QoS 保证，近来提出了许多新方法. 这些方法大致分为 2 类：（1）基于资源预留：网络资源按照某个业务的 QoS 要求进行分配，制定资源管理策略. 互联网工程组 IEEF 提出的综合服务体系结构便是基于这种策略，资源预留协议 RSVP 是其核心部分. （2）基于优先级：网络边界节点对业务流进行分类、整形、标记. IEEF 提出的区分服务便是基于这种策略[1].

在综合模型中，一般要求网络中所有的路由器都有 RSVP 模块，才能很好地支持 QoS. 如果有不支持 RSVP 的路由器在网络中工作，它们随时都可能降低用户所期望的 QoS，就会使端到端的 QoS 得不到保证. 但是目前仍有不支持 RSVP 的路由器在 IP 网上使用，为了解决这一问题，本文提出了利用隧道 agent 这一移动 agent 技术的解决方案.

2　隧道 agent

如图 1 所示，在数据流传送的路径上，把那些不支持 RSVP 的路由器所组成的集合称为隧道，把不支持 RSVP 的路由器称为隧道个体，把与隧道相邻的支持 RSVP 的路由器称为边缘路由器，隧道 agent 被放置在每个隧道的边缘路由器上，用它来监视隧道的性能，并与支持 RSVP 的路由器交互以保证端到端的 QoS[2].

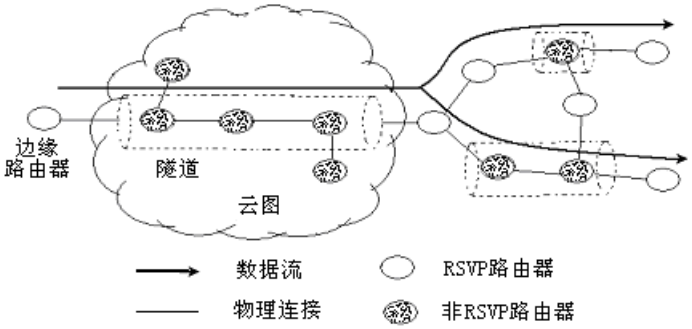

图 1　隧道 agent

收稿日期：2005－12－21
作者简介：许华荣（1970－），男，福建莆田市人，讲师，博士生.

3 隧道的建立

假设每个支持 RSVP 的路由器不但可以执行隧道 agent，而且，当已建立隧道失效后，它也能执行重建隧道的代码. 重建隧道要尽可能的利用旧隧道的连接资源，同时满足用户要求端到端的 QoS 保证（要求的带宽和最小延迟）. 建立新隧道的方法是产生一组移动 agent，每个节点上都有一个移动 agent,每个移动 agent 被设置有两个感知模块，一个用来感受它的邻居环境，一个能按移动规则决定它的移动. 重建隧道从父边缘路由器开始遍历网络各节点，根据各节点的邻近的节点的状态，选择它的下一个要连接的节点，选择的条件由下面两个方程决定 2.1 中的方程 1 决定 ，当找到目标节点后，就确立了新的隧道，隧道 agent 将从旧的隧道边缘移向新的隧道边缘.

3.1 移动的概率分析

一个移动 agent(k)在节点 i,在时间 t 内它移向节点 j 的概率为：

$$P_{ij}^{k}(t)=\frac{[OF_j(t)]^{-\Phi}[T_{ij}(t)]^{\alpha}[C_{ij}(t)]^{-\beta}}{N_k} \qquad \text{----------} (1)$$

$$N_k=\sum_{j\in(S(i)-Tabu(k))}[OF_j(t)]^{-\Phi}[T_{ij}(t)]^{\alpha}[C_{ij}(t)]^{-\beta} \qquad \text{---------} (2)$$

其中 OF 是 j 节点所占有的队列大小，T_{ij} 表示从节点 i 到节点 j 的物理相连信息,C_{ij} 代表从节点 i 到节点 j 的延迟大小,α ,β ,Φ 是常数，是代表各自节点对延迟和占有队列的敏感程度. N_k 是简单的标准化因子,S(i)节点 i 的邻居节点集合，tabu(k)是移动 agent（k）已访问的节点的集合. 从以上方程可以看出，移动 agent 尽量选择数据流少的节点，以达到平衡负载的目的.

3.2 隧道建立算法

如图 2 所示，有一数据流从边缘路由器 A 经过 B，C，D 和 E 然后到达 F，路由器 C，D，E 组成了隧道，隧道 agent 放置在 B 路由器上.

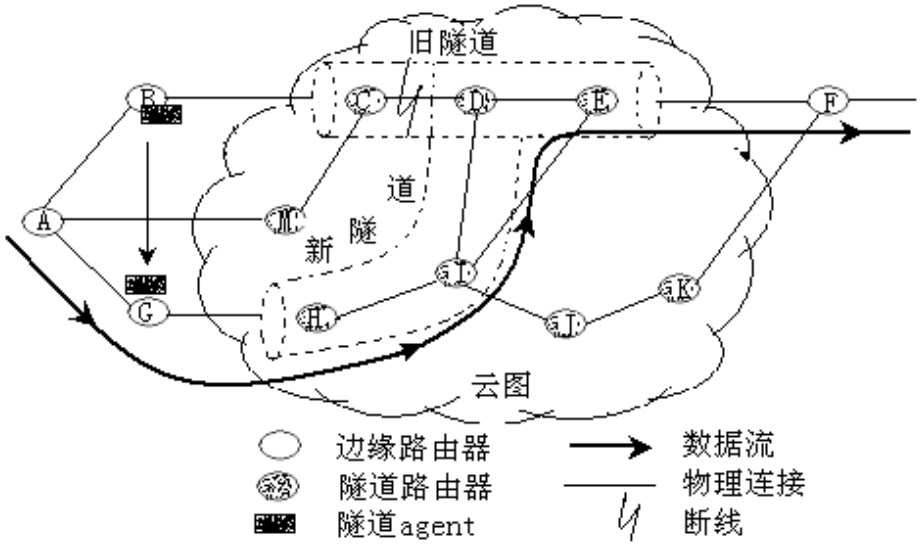

图 2 隧道重建过程图

当路由器 C 和 D 的连接失败，在 B 路由器上的隧道 agent 发现它不是在一个有效的隧道的边缘，就运行重置程序，搜索一条新的可用的路径，并把隧道 agent 迁移到新的隧道的边缘，如图 2 所示，迁移到 G 路由器上同，而新的有效的隧道是 H，I，D，E. 其中路由器 D 和 E 是旧隧道的节点.

假设隧道 agent 在节点 i（如图 2 的节点 B），第 k 个节点（如图 2 的节点 C）与节点 D 的连接失败，而不再是有效隧道的节点. 因此就必须重新建立隧道以实现最佳的资源预留路径.

算法从父边缘路由器开始（路由器 A）,产生一组移动 agent，算法的步骤如下：

假设 n 是总的节点数，Tmax 是总的模拟时间，τ 是移动 agent 进入并建立的时间.

```
Initialization()
Set time t=0
Do while(t<=0)
{ Generate a population of n agents at the source node
   Repeat{
        For ( each k agent from 1 to n)
          {
              The agent selects its next non-visited neighboring tunnel node j with a selection probability
                   function Pij, then push this node in the agent stack memory(tabu-list(k)).
              if( reaching a node where no un-visited nodes with sufficient link BW can be founded)
              then incrementing the sleeping counter of agent k
           else Push this node in agent stack memory ,Delay(k)=Delay(k)+delayij
           if ( Delay(k)>Tmax_Delay)
           then kill agent(k)
           if ( the sleeping counter of agent k >Max_sleeping_threshold)
           then kill agent(k)
           if ( any agent reached the destination node)
              {-Generate a backward agent
               -pop nodes from the stack memory of the forward agent(opposite order of visiting)
               -for(each popped node from stack)
                   { let the backward agent modify the pheromone level on the rout according to the rout
                        quality}
             }}   // end for k
    t=t+ τ
} // End   Do-while
```

上述算法的目标是建立一个新的隧道，隧道建立后隧道 agent 就迁移到新的边缘路由器上（如图 2 G 的节点）.

4　隧道 agent 与 RSVP 的交互

标准的 RSVP 具有检测不支持 RSVP 路由器的能力，当它检测到一个隧道路由器时，会激发其中的隧道 agent 来初始化隧道设置. 方法是利用接收方的程序发送的探测 agent 来探测. 在探测到了边缘路由器后，隧道 agent 被复制到边缘路由器上执行，隧道 agent 在边缘路由器上主要工作就是启动 traceroute 程序来获得所有的隧道路由器信息（同是也通过 SNMP 的 MIB 得到丢包率、队列长度等参数），返回给下行的隧道 agent.

端到端的 QoS 的保证是要求端到端的传输延迟为一个估定的值，假设为 τ，分组所经过的节点数为 n，那么只要每个 RSVP 路由器预留资源的平均传输延迟不大于 τ/n，就可以得到有保证的实时通信. 但是在有不支持 RSVP 的路由器存在的情况下，因其延迟是不可预测的，所以就不能对每个路由器分配平均的延迟时间以保证传输路径上的总的延迟时间 τ. 解决这个问题的关键是预测不支持 RSVP 路由器的传输延迟，再通过减少支持 RSVP 路由器上的平均延迟时以弥补不支持 RSVP 的路由器上产生的大量延迟. 这时可以利用移动 agent 从一端发出一个带有时间戳的消息，然后在另一端接收消息，计算它的延迟，假设为 τ_1，其中假设支持 RSVP 的路由器为 n_1，这样只要每个 RSVP 路由器新的资源预留平均延迟不大于（$\tau - \tau_1$）/ n_1，就可以得到有保证的实时通信. 如果 τ_1 比 τ 还大，这时可以提高 τ 以降低服务质量，或者就中止 RSVP 的资源预留. 随着时间的推移，不支持 RSVP 的路由器的延迟可能改变，所以移动 agent 要监视它们的状态，以判断能否保证实时通信. 不能保证，则要调整 RSVP 的路由器的新的资源预留的平均延迟[4].

5 结束语

本文提出利用隧道 agent 实现在有不支持 RSVP 的路由器在网络中工作情况下的资源预留，以保证端到端的 QoS. 设计了建立隧道的算法，隧道建立后通过隧道 agent 与 RSVP 的交互实现在整个数据传输路径上的资源预留. 隧道 Agent 具有移动 agent 的特性，能够实时地监视网络的工作情况，并对 QoS 参数作出相应的调整，它是 IP 网上实现可靠的端到端 QoS 的可取的方案，如何对网络设备进行相应的改造，使它们能支持移动 agent 的运行，从而实现高效而可靠的 IP QoS 控制将是今后研究的重点.

参考文献：

[1]李学军，李 洪，朱英军，宽带 IP 城域网的优化策略下实践，人民邮电出版社，2002.12.
[2]雷振甲，计算机网络管理及系统开发. 北京：电子工业出版社,2002.
[3]林 闯，多媒体信息网络 QoS 的控制. 软件学报，1999.10 (10)，1017-1019.
[4]张云勇，移动 agent 及其应用，清华大学出版社 2002.1.
[5]Antonio Puliafito,Orazio Tomarchio,Lorenzo Vita,Design and Implementation of a Moblie Agents' Plateform[J],Joumal of System Architecture ,2000,46:145-162.