WDM网络中满足延迟和延迟差约束的分布式组播路由与波长分配算法

黄传河　陈莘萌　贾小华　张文涛
(武汉大学计算机学院,武汉 430072)
E-mail:hwanghe@public.wh.hb.cn

摘　要　在WDM网络中,由于每条链路上可用波长是动态变化的,在考虑波长转换延迟时间的条件下,实现实时组播连接的路由与波长分配是十分困难的。论文提出了一种用于建立满足延迟时限和延迟差要求的实时组播连接的分布式路由与波长分配算法。该算法假定每个节点没有全局路由信息,只根据关联链路的信息进行路由选择,且将路由与波长分配统一进行。组播路由算法以Prim最小生成树算法为基础,生成一棵满足给定延迟时限的最小成本树。对不满足延迟时限的目的节点,通过增加回路边构造回路再消除长延迟路径的方式,加入到组播树中。对不满足延迟差的目的节点,采用重构Steiner树的方法,使其满足延迟差的要求。波长分配使用最少波长转换和负载平衡策略。

关键词　路由与波长分配　组播路由　延迟限制路由　延迟差限制路由

文章编号 1002-8331-(2003)22-0168-06　**文献标识码** A　**中图分类号** TP393

A Distributed Routing and Wavelength Assignment Algorithm Satisfying Delay and Delay Variation Bound for Multicast in WDM Networks

Huang Chuanhe　Chen Xinmeng　Jia Xiaohua　Zhang Wentao
(Computer School, Wuhan University, Wuhan 430072)

Abstract: Routing and wavelength assignment for online real-time multicast connection setup is difficult due to the dynamic change of availabilities of wavelengths on links and the consideration of wavelength conversion delay in WDM networks.This paper presents a distributed routing and wavelength assignment algorithm for the setup of real-time multicast connections satisfying delay bound and delay variation bound.The algorithm assumes that each node does not have point-to-point routing information.It selects routes according to incident links only and integrates routing and wavelength assignment as a single process.The method is to add a minimal cost link to the tree at a time according to MST algorithm.For destinations not included in the tree,they are added to the tree by adding a link to the tree to make a cycle then to remove a longer delay path.Then delay variation is checked.If it is not satisfied,then the Steiner tree is reconstructed,and a longer-delay path is selected to replace the original low-delay path for the destinations not satisfying delay variation bound.The algorithm assigns wavelength according to the rule of least-conversion and load-balance.

Keywords: Routing and Wavelength Assignment, Multicast Routing, Delay Bound Routing, Delay Variation Bound Routing

1　引言

WDM(wavelength division multiplexing)将光纤的带宽分为互不重叠的并行通道，每个通道使用一个波长传输信号。WDM是充分利用网络带宽的关键技术。WDM网络采用面向连接的方式工作,在数据传输前,必须在通信双方之间建立连接。在WDM网络中建立连接包括路由选择和波长分配两个过程,简称为RWA(Routing and Wavelength Assignment)。

组播是一种组通信机制,其发送者(源节点)将消息同时发送给一组接收者（目的节点)。实时组播是一类特定的组播形式,要求在组播请求到达后尽快建立组播连接,并且每个节点通常不具有全网完整准确的全局状态信息。实时组播在现代计算机网络中有广泛的应用,例如电视会议、多媒体教学、视频点播(VOD)、网上拍卖等。

建立实时组播的路由就是找到一棵以源节点为树根、包含所有目的节点的树,称为组播树。在WDM网络中,组播树的每条边对应一条光纤链路,需要分配一个可用波长。当节点没有波长转换器时,从树根到树叶的路径上的每条光纤链路需要分配同一波长。共享同一链路的不同组播树,在共享链路上需要分配不同的波长。这样的组播树也称为波长路由树。实时组播应用通常要求从源节点(树根)到任一目的节点(树叶)的延迟时间不超过给定的时限,同时,从源节点到任意两个目的节点的延迟差不超过给定的限制值，在满足这些约束的条件下,使得组播树的总成本最小。业已证明,寻找这种组播树的问题是NP-hard问题,因此只能应用启发式方法找到接近最优的满意解[1,2]。

论文提出了一个在WDM网络中建立满足延迟时限和延

作者简介:黄传河,副教授,博士,主要研究方向:计算机网络、分布并行处理。陈莘萌,教授,博导,主要研究方向:分布并行处理,新型计算机理论。贾小华,教授,博导,主要研究方向:计算机网络、分布并行处理。张文涛,硕士研究生,研究方向:计算机网络。

迟差要求的实时组播连接的分布式路由与波长分配算法。该算法首先构造一棵最小成本树连接满足延迟时限的目的节点。对没有连入最小成本树的节点，利用构造并消除回路、降低路径延迟的办法将其余目的节点加入到组播树中。对不满足延迟差要求的目的节点，重构 Steiner 树，使其满足延迟差的要求。该算法的优点是：

(1)它是完全分布式的，路由与波长分配只根据本节点的信息完成。

(2)在建立组播连接时尽量避免波长转换，达到负载平衡。

(3)所构造的树在满足延迟时限和延迟差要求的条件下具有接近最优的成本。

2 问题定义

假定：

(1)组播树除了需要满足延迟时限的要求外，还需要满足延迟差的要求。延迟差(Delay Variation)定义为从源节点到任意两个目的节点的延迟时间的差值。

(2)每个节点都没有保存点到点的路由信息，而只有有关相邻节点和关联链路的信息。同时目的节点集较大，适于规模较小的网络或者目的节点集包含很多节点（与网路规模相比）的情况。

(3)每个节点具有一个全波长转换器，即可以将任一波长转换为任一其它波长，所有节点完成任何两个波长之间的转换所需时间是相同的。

具体定义为：

网络用无向图 $G(V,E)$ 表示，其中 V 为节点集，E 是光纤链路集。每条链路 (i,j) 有 3 个参数：

- $\sigma_{ij}\subseteq\{1,2,\cdots,W\}$ 表示链路 (i,j) 上当前可用波长的集合。
- c_{ij} 表示使用链路 (i,j) 的成本。
- d_{ij} 表示链路 (i,j) 的延迟时间。

链路 (i,j) 上的可用波长是动态变化的，只有与此链路相连的两个节点确切知道当前 σ_{ij} 的值。在建立连接时，为该连接的每一链路分配一个当前未被使用的波长。被分配的波长一直被占用，直到通信结束连接被终止。

在一条路径上的通信延迟包括链路延迟和波长转换时间两部分。链路延迟 d_{ij} 表示信号从节点 i 经链路 (i,j) 到达节点 j 所需的时间。波长转换时间 $d_i^c(\lambda_x,\lambda_y)$ 表示节点 i 将波长 λ_x(输入波长)转换为波长 λ_y(输出波长)所需的时间。如果没有进行波长转换，即 $\lambda_x=\lambda_y$，则 $d_i^c(\lambda_x,\lambda_y)=0$。

考虑建立组播连接的实时请求 $R=(s,D,\Delta,\delta)$，其中 s 是源节点，D 是目的节点集合，Δ 是延迟时限值，δ 是延迟差(见下述定义)。组播连接是一棵树 T，T 的总成本定义为：

$$COST(T)=\sum_{(i,j)\in T}c_{ij} \tag{1}$$

令 $P(u,v)$ 表示树 T 中从节点 u 到节点 v 的路径，该路径的成本定义为：

$$COST(u,v)=\sum_{(i,j)\in P(u,v)}c_{ij} \tag{2}$$

在树中从节点 u 到 v 的延迟 $DELAY(u,v)$ 定义为：

$$DELAY(u,v)=\sum_{(i,j)\in P(u,v)}d_{ij}+\sum_{(i,j)\in P(u,v)}d_i^c(\lambda_x,\lambda_y) \tag{3}$$

从树根 s 到任意节点 v 的延迟记为 $DELAY(s,v)$，树 T 的延迟定义为：

$$DELAY(T)=MAX\{DELAY(s,d),\forall d\in D\} \tag{4}$$

延迟时限条件可以表述为：

$$DELAY(T)\leq\Delta \tag{5}$$

δ 为组播应用允许的延迟差，需要满足的条件是：

$$|DELAY(s,d_1)-DELAY(s,d_2)|\leq\delta,\forall d_1,d_2\in D \tag{6}$$

论文所考虑的问题是设计一个分布式的路由与波长分配算法为实时组播请求构造一棵波长路由树，使得该树在满足(5)和(6)定义的条件下成本尽可能小。

3 有关研究结果

目前已提出的在 WDM 网络中实现组播的路由与波长分配的主要研究结果有：

光树方法 (Light-Tree)：L.H.Sahasrabuddhe 等提出了光树(Light-Tree)的概念[3]，其思想是预先定义一些树，为其分配波长。一旦有组播请求，就将其适配到预定义的光树中。但因为组播请求的不确定性，预定义的光树并非总能容纳任意的组播请求，同时波长的利用也是一个需要解决的问题。

光迹方法：M.Ali 等提出了一种用光迹(Light-Trail)代替光树的想法[4]，其方法是从源节点出发，沿链路搜索，每个链路只允许经过一次，但节点允许经过多次，最后找到一条路径，包含所有目的节点。该方法对节点没有光分裂功能的网络是有效的，但信息从源节点到达目的节点需要经过很长的路径(亦即时间)。

优化方法：X.Jia 等提出了一个为一组静态组播请求分配路由和波长的静态算法[5]，其方法是构造一组树，并用图着色方法分配最少数量的波长，然后进行优化。N.Sreenath 等提出了一种基于虚拟源的组播路由思想，其方法是将节点按优先级分类，优先考虑高优先级节点加入到树中，当低优先级节点不能加入到树中时，构造一棵新树[6]。

受限组播的实现方法：WDM 网络有时存在一些特定的限制条件，如信号能量限制即信号经过的链路或节点数不能超过一定的阈值，每个节点光的可分裂数量、延迟限制等。X.Zhang 等提出了构造光树的四种方法[7]，分别是：(1)re-route-to-source，先生成组播树，再检查分枝节点是否超过分枝数。若超过，将该分枝加到往树根方向找到的第一个满足条件的节点。(2)re-route-to-any，方法同(1)，只是被剪枝的分枝可加入到任何节点。(3)member-first，生成树时优先加入 member 节点(用 Dijkstra 算法)，然后再检查分枝点。(4)member-only，生成树时只加入 member 节点。V.Gupta 等提出了在不能建立一棵树时建立森林来实现组播的思想，并提出了将树分解为森林的方法[8]。

大量的组播应用通常附带有服务质量(QoS)的要求。例如最大延迟时间、延迟差、带宽限制、成本等，或它们的组合。上述所有方法并没有真正解决大型网络中具有服务质量要求的动态组播的路由与波长分配问题。

4 分布式组播路由与波长分配算法 vdRWA

4.1 数据结构

可用波长链路数 $NL(\lambda)$：每个节点记录与其关联的链路上的可用波长。$NL(\lambda)$ 表示与本节点相关联的链路中波长 λ 可用的链路总数。

前驱节点 $P(v)$：表示树中节点 v 的前驱节点(父节点)。

后继节点 $S(v)$：表示树中节点 v 的后继节点(子节点)，可

能不止一个。

允许的输出节点集合 Allowed[d]:从本节点出发到达目的节点 d 允许经过的直接输出节点(下一个节点,亦即子节点)的集合。通常是人为指定的,以指出到达目的地的方向,减少网络中路由选择的盲目性。

4.2 基本思想

论文所提出的分布式路由与波长分配算法 vdRWA 实现的功能是建立实时组播连接,即为实时组播请求建立一棵波长路由树。算法 vdRWA 分三个阶段完成:第一阶段调用函数 GenTree 构造一棵最小成本树,其中每个目的节点 d 都满足延迟时限(5)的要求。如果所构造的树没有包含所有目的节点,则进入第二阶段,调用函数 Extend 将其余的目的节点加入到树中。如果 Extend 不能将所有节点加入到树中,说明延迟时限值 Δ 太小,算法失败。如果包含所有目的节点的树同时满足延迟差的条件(6),则算法正常终止。否则,进入第三阶段。第三阶段调用函数 Repair 对所构造的树进行修正,使其满足延迟差条件(6)。

4.3 构造初始树——GenTree

(1)GenTree 基本思想

由于每个节点都没有关于全网的状态信息,因此不可能采用 Dijkstra 算法或其它类似算法直接计算最短路径,而只能通过逐节点地试探,选取最小成本同时满足延迟时限要求的节点来扩展生成的组播树。GenTree 采用 Prim 最小生成树 MST 算法的思想,从源节点开始,一次扩展一个节点,直到所有目的节点加入到树中。主要步骤为:

①将源节点加入到树中。

②每次挑选一个新的不在树中的节点加入到树中。挑选方法是:由源节点向树中每个节点发送一个 FIND 消息,其中包括未加入到树中的目的节点集 D'、不在树中的节点集 T'。树中的所有节点(称为挑选节点)将成本最小且满足延迟要求的、未包含在树中的邻节点(称为候选节点)通过 FOUND 消息告诉源节点。挑选节点 v 选择候选节点 w 的标准为

$$f(v,w)=\begin{cases}c_{vw}, DELAY(s,w)\le\Delta, |\sigma_{vw}|\ge 1, w\notin Tree, w\in T' \\ \infty, \quad otherwise\end{cases} \tag{7}$$

$$DELAY(s,w)=DELAY(s,v)+DELAY(v,w)+d_v^c(\lambda_v,\lambda_w) \tag{8}$$

由于每次只加入一个节点到树中,因此,每次挑选候选节点时,并不要求每个挑选节点重复发送 FOUND 消息,因为源节点可以记录每个 FOUND 消息,只有新加入的节点和最小输出链路因树的扩展而改变的挑选节点才需要发送 FOUND 消息给源节点。

③源节点收到挑选节点送回的 FOUND 消息后,选择一个成本最小的候选节点即 $f(v,w)$ 最小的节点 w 加入到树中。如果有多个候选节点的 $f(v,w)$ 相同,则优先选择是目的节点的候选节点加入到树中。如果候选节点都是目的节点,则优先选择离树根近(延迟小)的节点。如果离树根的距离(延迟)相同,则从中任选一个节点加入到树中。

④源节点向挑选节点及被选中的候选节点发送 ADD 消息,将候选节点加入到树中。挑选节点将新加入的链路 (v,w) 标为树边(Tree Link)。

⑤选中的挑选节点(新加入的节点的父节点)负责为新加入的链路分配波长。

⑥挑选节点计算从源节点到候选节点的延迟时间并将其附在 ADD 消息中转发给候选节点。

⑦候选节点接收到 ADD 消息后记录延迟时间值,并将前驱链路标为反向边(Reverse Link)。

⑧新选中的候选节点通知在树中的邻节点记录 CYCLE-LINK(回路链路)信息,即如果 v 是新加入到树中的节点,v 的邻节点 w 已包含在树中,则 w 将链路 (w,v) 记为 CYCLE-LINK。因为将 (w,v) 加入到树中将产生回路。

⑨重复上述过程②~⑧直到所有目的节点都加入到树中。

⑩删除树中多余的节点和链路(树叶不是目的节点的节点及其输入链路)。

(2)GenTree 的波长分配策略

由于假定每个节点都具有波长转换功能,因此每个节点原则上都可以为新加入的链路分配任意可用波长。但是,因为波长转换产生延迟,过多的波长转换可能导致过长的延迟从而无法生成满足延迟要求的树,因此在分配波长时需要考虑波长转换。另外,好的分配策略能为后来的请求留出更多可用波长,可以满足更多的组播请求,提高网络的吞吐量。

GenTree 分配波长的算法是:

①源节点为第一条链路分配具有最大 $NL(\lambda)$ 值的波长。

②在每个中间节点,优先选择与输入链路上相同的波长进行分配。

③当输出链路上与输入链路上相同的波长不可用时,选择具有最大 $NL(\lambda)$ 值的波长进行分配。

4.4 扩展树包含所有目的节点——Extend

(1)Extend 基本思想

一个目的节点不能在第一阶段通过 GenTree 加入到树中的主要原因是所选链路没有可用波长,或者虽有可用波长,但路径延迟超过给定的延迟时限。对那些没有加入到树中的目的节点的处理方法是,试图选取链路不在树中但两个端点已在树中的链路加入到树中,使树中包含回路。然后消去先前加入的部分链路,消除回路,以降低树或部分路径的延迟时间,从而为其余的目的节点加入树中、形成满足延迟时限条件(5)的树提供条件。

Extend 的基本思想就是采用这种制造并消除回路的方法,对已经生成的树进行链路替换,以降低树中某些路径的延迟,使得还没有加入到树中的目的节点能够加入到树中。

在构造并消除回路、进行链路替换的时候,应遵循下述基本规则:

①应尽可能多地降低树或相关路径的延迟时间。

②应尽可能使尽量多的节点降低延迟时间。

③应有利于扩展树包括其余的目的节点,即延迟时间降低了的节点应该离其余未加入到树中的目的节点更近,以使它们易于加入树中。其量化指标是从降低了延迟的树节点通向其余目的节点的链路应尽可能多地不在树中,或者虽在树中,但不是树中的反向边(从树叶到树根)。

为满足上述要求,当选择回路边时,应计算:

①树或路径降低的延迟时间值;

②延迟时间降低了的树节点的数量;

③因增加特定的回路边到树中而使剩余的目的节点离树更近的节点数。

但上述③的数量因为缺乏全局信息和不确定的波长转换而通常不可能计算,因此可以只计算①和②。以哪种顺序作为选择的标准,反映了不同的目标,通常其结果可能会有所不同。如果以①为主要标准,可能会使树的一个分枝增长更快。如果以②为主要标准,可能使得剩余的目的节点有更多的可选路径

加入到树中。

算法的主要步骤是(主要参数放在消息名称后的括号内):

①源节点沿树边向树中每个节点发送 CYCLE-FIND(D′)消息;

②每个关联 CYCLE 边的节点 v 计算经每条 CYCLE 边(v,w)到邻节点 w 的延迟时间 $C\_DELAY(s,w)=DELAY(s,v)+d_{v,w}+d_v^c(\lambda_{P(v),v},\lambda_{v,w})$。向 w 发送 *CYCLE-TEST* 消息以测试并确定回路的起始节点,同时沿 *Tree* 边转发 CYCLE-FIND 消息。

③关联 CYCLE 边的树节点 w 收到 CYCLE-TEST 消息后,比较 C_DELAY(s,w)和 DELAY(s,w)。如果 DELAY(s,w)>C_DELAY(s,w),则

(a)令 v 为回路起始节点(CSN),w 为回路终止节点(CEN);计算经 CYCLE 边到达 w 降低的延迟 δr=DELAY(s,w)-C_DELAY(s,w)和到达 w 的原前驱节点的延迟时间 C_DELAY$(s,P(w))$=C_DELAY$(s,w)+d_w^c(\lambda_{v,w},\lambda_{w,P(w)})+d_{w,P(w)}$;

(b)沿 w 的每条 Tree 边(w,u),计算降低的延迟时间 δt=DELAY$(s,w)+d_w^c(\lambda_{P(w),w},\lambda_{w,u})$-C_DELAY$(s,w)-d_w^c(\lambda_{v,w},\lambda_{w,u})$;如果 $\delta t>0$,沿 Tree 边发送 REACH_TEST(D',δt,CSN,CEN)消息,以通知其余的后继树节点修改延迟;

(c)如果 $\delta r>0$,沿 Reverse 边发送 R_REACH_TEST(D′,δr,CSN,CEN)消息,以通知原前驱树节点修改延迟;

(d)向源节点发送 REACH_REPLY(CSN,CEN,max(δt,δr))消息;

否则,如果 DELAY(s,v)>DELAY$(s,w)+d_{w,v}+d_w^c(\lambda_{P(w),w},\lambda_{w,v})$,

(e)令 w 为回路起始节点(CSN),v 为回路终止节点(CEN),计算经 CYCLE 边(w,v)到达 v 的延迟 C_DELAY(s,v)=DELAY$(s,w)+d_{w,v}+d_w^c(\lambda_{P(w),w},\lambda_{w,v})$;

(f)向 v 发送 CYCLE_REPLY(CSN,CEN,C_DELAY(s,v))消息;

④节点 v 收到 CYCLE_REPLY 消息后,

(a)计算降低的延迟时间 δr=DELAY(s,v)-C_DELAY$(s,v)-d_w^c(\lambda_{w,v},\lambda_{v,P(v)})$;

C_DELAY$(s,P(v))$=C_DELAY$(s,v)+d_v^c(\lambda_{w,v},\lambda_{v,P(v)})+d_{v,P(v)}$;

(b)对每一条 Tree link(v,u),

(i)计算 δt=DELAY$(s,v)+d_v^c(\lambda_{P(v),v},\lambda_{v,u})$-C_DELAY$(s,v)-d_v^c(\lambda_{w,v},\lambda_{v,u})$;

(ii)如果 $\delta t>0$ 则沿树边方向转发 REACH-TEST(D',δt,CSN,CEN)消息;

(b)如果 $\delta r>0$ 则沿 Reverse link(向树根)方向发送 R-REACH-TEST(D',C_DELAY$(s,P(v))$,CSN,CEN)消息;

(c)向源节点发送 REACH-REPLY(CSN,CEN,max(δt,δr))消息;

⑤收到 REACH_TEST 消息的节点 u,对每条树边(u,q),

(a)沿树边转发 REACH-TEST 消息;

(b)计算并修改延迟时间 DELAY(s,u)=DELAY$(s,u)-\delta$;

(c)向源节点发送 REACH-REPLY(CSN,CEN,δ);

⑥收到 R_REACH_TEST 消息的节点 u(消息来自节点 v),

(a)计算降低的延迟 δ=DELAY(s,u)-C_DELAY(s,u);

(b)如果 $\delta>0$ 则向源节点发送 REACH-REPLY(CSN,CEN,δ)消息;

(c)对每条 Tree 边(u,w)(边(u,v)除外),

(i)计算 δt=DELAY$(s,u)+d_u^c(\lambda_{P(u),u},\lambda_{u,w})$-C_DELAY$(s,u)-d_u^c(\lambda_{v,u},\lambda_{u,w})$;

(ii)如果 $\delta t>0$ 则沿 Tree 边发送 REACH-TEST(D',δt,CSN,CEN)消息;

(d)对 Reverse 边,

如果 C_DELAY$(s,u)+d_u^c(\lambda_{v,u},\lambda_{u,P(u)})$<DELAY$(s,u)$即可降低延迟,则

(i)计算 C_DELAY$(s,P(u))$=C_DELAY$(s,u)+d_{u,P(u)}+d_u^c(\lambda_{v,u},\lambda_{u,P(u)})$;

(ii)沿 Reverse 方向发送 R-REACH-TEST(D',C_DELAY$(s,P(u))$,CSN,CEN)消息;

⑦源节点收到 REACH_REPLY 消息后,为每条 CYCLE 边计算降低的延迟时间值、延迟降低了的节点的数量,从所有 CYCLE 边中选择最佳的一条 (x,y),向节点 x 发送 CYCLE_ADD 消息,通知将(x,y)加入到树中;

⑧节点 x 将(x,y)标为 Tree 边;节点 y 修改延迟时间,将(y,x)标为 Reverse,沿原 Reverse 边向原前驱节点发送 CYCLE_BREAK 消息,沿 Tree 边发送 DELAY_RECOMPUTE 消息,通知后继节点重新计算并修改延迟时间,将原 Reverse 边标为 Tree;

⑨收到 DELAY_RECOMPUTE 消息的节点重新计算延迟时间,如果延迟满足时限要求,则转发 DELAY_RECOMPUTE 消息,否则,源节点发送 DELETE_NODE 消息,将 Reverse、Tree 边标为 Unknown,向后继节点发送 DELETE_LINK 消息删除后继节点。

⑩收到 CYCLE_BREAK 消息的节点 w(消息来自节点 u),如果 C_DELAY (s,w)<DELAY (s,w),则沿每条 Tree 边发送 DELAY_RECOMPUTE 消息,沿 Reverse 边发送 CYCLE_BREAK 消息通知原前驱节点删除回路。否则,将(w,u)标为 Unknown,向 u 发送 CYCLE_STOP 消息,通知 u 将(u,w)标为 Unknown,回路消除,(u,w)为删除的边。

(2)Extend 的波长分配策略

Extend 所用的波长分配策略与 GenTree 所用的策略基本相同。

GenTree 分配波长的算法是:

①回路起始节点,优先选择与输入链路上相同的波长进行分配。

②所选回路上与输入链路上相同的波长不可用时,选择具有最大 $NL(\lambda)$值的波长进行分配。

4.5 修正树满足延迟差要求——Repair

(1)Repair 基本思想

满足延迟时限要求的最小成本树不能满足延迟差的要求的原因是有些路径的延迟小,而有些路径的延迟太大。路径延迟太大的原因可能是所经历的链路的延迟时间太长,或者是经历太多的波长转换,或者二者兼而有之。

一种解决办法是降低长延迟路径的延迟,即重新分配波长以减少波长转换,或重选延迟较小的链路。这种方法通常难以

实现，因为在分配波长时已经考虑了相邻链路的波长转换问题，现在只有对整个路径上的波长进行考察才知道是否存在过多的波长转换，而这一过程是费时的，并且并不能保证有合适的可供选用的波长用于分配。而选择小延迟的链路通常也十分困难，而且还存在一种可能，就是通常延迟小的链路速度就快，成本也就更高。

另一种解决办法是适当增加小延迟路径的延迟。可以重新分配波长故意增加波长转换而增加路径延迟，也可以重选延迟较长的路径。这种方法通常更可能达到目的。Repair 将以这种方法为主。

Repair 所采用的方法的基本思想为：

①源节点向所有树节点发送 DISCONNECT 消息，相关节点收到消息后将目的节点及其输入链路从树中删除。

②用 GenTree 和 Extend 的方法将所有 Steiner 节点（非目的节点）加入树中。

③源节点向所有树节点发送 FIND-DEST 消息，树节点返回一个可连接的目的节点。

④源节点选择满足延迟时限的具有最长路径延迟的一个目的节点加入树中。

⑤重复步骤④选择其余的目的节点加入树中。这时选择新的目的节点的标准及其优先顺序为：

(1)路径延迟满足(5)；

(2)延迟差满足(6)并且尽量小；

(3)成本最小。

⑥如果没有目的节点满足条件，则选择满足延迟时限(5)并且使得延迟差最小的链路加入到树中。

⑦如果所有目的节点都加入到树中，则对多余的路径进行剪枝。否则，算法失败。

(2)Repair 的波长分配策略

Repair 使用的波长分配策略与 GenTree 所用策略相同。

4.6 算法分析

定理 1：算法 vdRWA 所生成的树包含源节点和所有目的节点，满足延迟时限(5)的要求，并且满足延迟差的要求或者具有最小的延迟差，同时树的成本接近于最小。

证明：当算法成功结束时，所生成的树显然包含源节点和所有目的节点。

GenTree 过程每次挑选最小成本且满足延迟时限的链路加入到树中，因此所生成的树满足条件(5)。

Extend 通过增加回路边的方式扩展树，通过增加回路边，使得原来的路径的延迟降低，从而最终包含更多的直至所有的目的节点，增加的每个目的节点都满足(5)的要求。

Repair 从已经构造的树中删除目的节点，添加 Steiner 节点，试图增加其它路径代替原来短延迟路径，但每一步都满足(5)的要求。因此最终所得到的组播树满足条件(5)。

Repair 在对树进行修正时，采用的是重新选取较长延迟的路径替换原来短延迟的路径，因此使得延迟差缩小。当最终不能满足(6)的要求时，Repair 选择延迟差小的路径构造树。因此所构造的树要么满足(6)，要么在不满足时，延迟差最小。

由于目的节点集接近于网络规模，每次挑选的又是成本最小的链路加入到树中，根据 Prim 算法可知，最终的树具有接近最小的成本。

定理 2：算法的最坏通信复杂性是 $O(n^3)$。

证明：GenTree 一次向树中节点广播一个 FIND 消息，树中各节点分别返回一个 FOUND 消息。每次的消息数最多分别是 $1,2,\cdots,n-1$。正常条件下这两类消息的总数为 $O(n^2)$。Extend 发送 CYCLE_FIND 消息后，收到的应答消息数最多为 $O(n)$，其它消息总数为 $O(n)$。由于 Extend 在 GenTree 中调用，因此 GenTree 和 Extend 的消息总数为 $O(n^3)$。Repair 只运行一次，使用 GenTree 和 Extend 生成中间树，消息数不超过 $O(n^3)$，因此总的通信复杂性不超过 $O(n^3)$。

4.7 模拟结果

论文模拟所用网络按[9]的方法生成。vdRWA 与 SPT、MST 算法进行比较。SPT 和 MST 用论文的方法进行计算，但不考虑延迟限制。模拟对象为组播树成本，它们针对 5 个参数即 Δ、延迟差 δ、$|D|$、波长可用性及相对波长转换时间进行模拟；组播连接建立时间，针对 $|D|$。

图 1 说明了组播树成本与延迟时限 Δ、延迟差 δ 之间的关系。波长转换时间设为平均链路延迟的 0.5 倍，在模拟过程中假定不考虑网络负载问题。最小延迟时限定义为 $\Delta_{min}=\max(\{DELAY(s,d),\forall d\in D\})$+平均转换次数×转换时间，$DELAY(s,d)$为从源节点 s 到目的节点 d 的最短路径延迟。平均转换次数设定为网络直径的一半。Δ 的值从 Δ_{min} 开始，每次增加 $\frac{\Delta_{min}}{10}$。SPT 和 MST 只受 δ 限制而不受 Δ 限制。SPT 和 MST 的曲线为常数，因为二者都不受 Δ 影响。论文算法的曲线介于二者之间。随着 Δ 的增加，更多的目的节点通过 MST 路径连入树中，导致树的成本下降。当 Δ 和 δ 足够大时，不会影响路由选择，最终的组播树变成 MST。

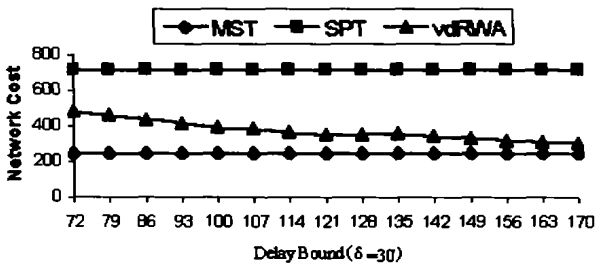

图 1 不同Δ时的成本(δ=30)

图 2 说明 vdRWA 所构造的组播树的成本与 δ、Δ 的关系。当 δ 较小时，δ 是制约成本的主要因素。当 δ 增大到一定值后，成本受 δ 的影响很小。

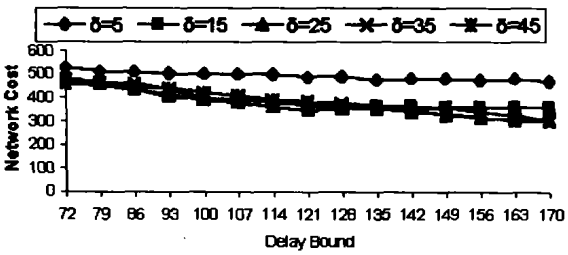

图 2 不同Δ和 δ 时的成本

图 3 说明树的成本与目的节点集大小间的关系。Δ 的值设为 $\Delta_{min}+\frac{\Delta_{min}}{4}$，δ 的值设为从源节点到各目的节点的最短延迟的最大差值。当目的节点增加时，组播树包括更多的目的节点，导

致树的成本增加。SPT 曲线在其它两个曲线之上并上升得更快。这是因为 SPT 不考虑路径共享。论文算法的性能接近 MST，二者的曲线随目的节点集的增大而增加得非常缓慢，因为目的节点集大，共享路径的可能性就大。

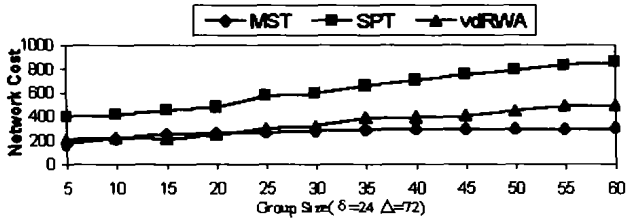

图 3　不同|D|时的成本

图 4~5 说明建立连接的成功率与 Δ、δ 的关系。当 δ 固定时，成功率随 Δ 的增大而增大。当 Δ 固定时(设为 $\Delta_{min}+\frac{\Delta_{min}}{4}$)，连接成功率随 δ 的增加而快速增大。论文算法的连接成功率明显高于 MST 和 SPT，这也是该算法的优点之一。

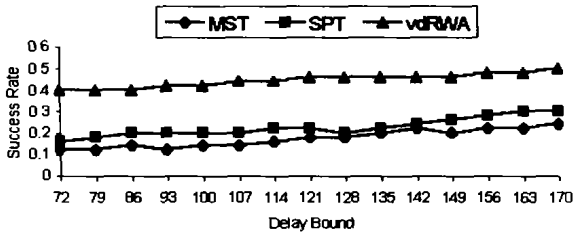

图 4　连接成功率与Δ的关系(δ=5)

5　结束语

论文提出的建立组播连接的分布式路由与波长分配算法，节点不需要保存任何全局信息，可以在满足延迟时限和延迟差的条件下，使所构建的组播树成本接近最小。该算法的不足之处是通信复杂性较高，当延迟时限和延迟差较严格时，建立连接的过程会较慢。对此需要做进一步的研究。

(收稿日期：2002 年 8 月)

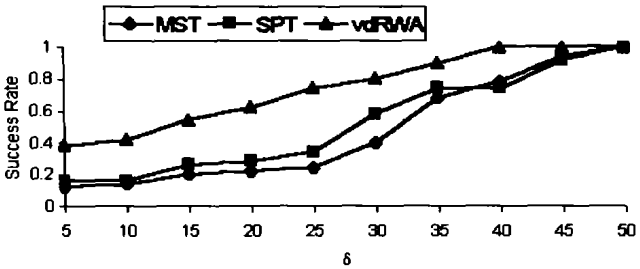

图 5　连接成功率与 δ 的关系(Δ=72)

参考文献

1.D Banerjee,B Mukherjee.Wavelength-Routed Optical Networks:Linear Formulation,Resource Budgeting Tradeoffs,and a Reconfiguration Study[J].IEEE/ACM Transactions on Networking,2000;(10):598~607

2.Ding-Zhu Du,J M Smith,J H Rubinstein et al.Advances in Steiner Trees[M].Kluwer Academic Publishers,2000

3.L H Sahasrabuddhe,B Mukhejee.Light-Trees:Optical Multicasting for Improved Performance in Wavelength-Routed Networks[J].IEEE Communications Magazine,1999;(2):67~73

4.M Ali,J Deogun.Cost-effective Implementation of Multicasting in Wavelength Networks[J].Journal of Lightwave Techlonogy,2000;(12):1628~1638

5.Xiao-Hua Jia,Ding-Zhu Du,Xiao-Dong Hu et al.Optimization of Wavelength Assignment for QoS Multicast in WDM Networks[J].IEEE Transactions on Communications,2001;(2):341~350

6.N Sreenath,K Satheesh,G Mohan et al.Virtual Source Based Multicast Routing in WDM Optical Networks[C].In:ICON2000,2000:385~389

7.X Zhang,J Wei,C Qiao.Constrained Multicast Routing in WDM Networks with Sparse Light Splitting[C].In:IEEE INFOCOM'00,2000:1781~1790

8.V Gupta,R Campbell.Reliable Sender-initiated Multicast for Improved QoS[C].In:Proc 9th ICCCN,2000:551~557

9.B M Waxman.Routing of multipoint connections[J].IEEE Journal on Selected Areas in Commun,1988;6(9):1617~1622

(上接 108 页)

底传中的概率为 0，F 会选择过人或直接射门。

(4)当 $R_F \to 0$ 时，$f \to 0$，此时 F 接近球门中央，F 可能直接射门。

(5)函数 f 的值主要和 θ_A，θ_F，R_A，R_F 有关，但同 R_E 也有很大关系。

通过函数 f，可以得到一个位于 0、1 之间的概率。在具体实现中将阈值定为 0.7，也就是说当返回值大于 0.7 时，防守队员 A 认为 F 有“下底传中”的意图，A 在做出对 F 的抢断决定的同时，利用“私语”通知 B 对 E 进行协防。这样做将有效地改进防守的质量。

3.2　抢断算法的改进

(2)式所描述的函数并没有考虑守门员的站位，实际上，在一些特殊情况下，守门员会脱离有效的控制范围，这时防守队员需要综合考虑后再决定是抢断还是补位。另外，许多传统的机器人足球队(如清华大学队)是采用神经网络技术来训练守门员，这也为综合算法增加了难度。总之，可以在对守门员及攻防队员位置比较的基础上改进抢断算法，论文就不再赘述。

4　小结

论文所讨论的部分基于动态目标驱动的抢断算法以“江大龙舟”机器人足球队(UJDB)为背景进行了模拟，并在 2002 年中国机器人竞赛等实践中加以检验，取得了较好的效果。作者今后工作的重点将放在针对各种进攻战术来完善抢断算法等方面，并在此基础上进行整体攻防策略的研究。作为一种典型的测试床，RoboCup 无论在理论和实践中都有广泛的探索空间，相应的研究工作必将推动 MAS 计算的不断完善与发展。

(收稿日期：2002 年 7 月)

参考文献

1.李毅等.基于 HIJ 的足球 KLMNO 和 PQRSTUUMV 平台[J].计算机研究与发展,2001;(8):911~915

2.吴建林等.基于案例学习和目标驱动学习的观点生成框架[J].系统工程理论与实践,1999;(2):1~7

3.项后军等.人工智能的前沿—智能体(Agent)理论及其哲理[J].自然辩证法研究,2001;(10):29~33