内容发布订阅系统路由算法和自配置策略研究*

薛 涛+, 冯博琴

(西安交通大学 电子与信息工程学院,陕西 西安 710049)

Research on Routing Algorithm and Self-Configuration in Content-Based Publish-Subscribe System

XUE Tao+, FENG Bo-Qin

(School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China)

+ Corresponding author: Phn: +86-29-82664160, E-mail: xt73@163.com

Received 2003-08-20; Accepted 2004-06-10

Xue T, Feng BQ. Research on routing algorithm and self-configuration in content-based publish-subscribe system. *Journal of Software*, 2005,16(2):251−259. http://www.jos.org.cn/1000-9825/16/251.htm

Abstract: Content-Based publish/subscribe systems have recently received an increasing attention. Efficient routing algorithms and self-configuration are two key issues in the area of large-scale content-based publish/subscribe systems. Although many routing algorithms have been proposed, none of them fully exploits multicast to enhance system performance and save network bandwidth. In addition, the vast majority of currently available publish-subscribe middleware has ignored this self-configuration problem. This paper first proposes a hierarchical system model with multicast clustering. Then a hybrid routing algorithm is presented, which can fully exploit multicast in order to reduce the used network bandwidth. Moreover, a multicast clustering replication protocol and a content-based multicast tree protocol are presented for coping with the node or link failures and rebuilding the event dispatcher trees. Experimental results reveal that the system has better routing efficiency and lower cost, and guarantees the self-configuration characteristic.

Key words: publish/subscribe; content-based routing; self-configuration; multicast clustering

摘 要: 路由算法和动态自配置特性是实现大规模基于内容的发布订阅系统的两个关键问题.尽管已经有多种路由算法被提了出来,但是它们没有充分利用组播技术提高系统性能和节省网络带宽;此外,已有系统的网络都是静态的,不能够进行网络的自动配置.首先,提出了具有组播集群的层次性系统模型,设计了混合式路由算法,充分利用物理网络组播的特性,节省网络带宽.然后,提出了组播集群复制协议和基于内容的组播树协议 CMTP,分别处理节点或者链路失效导致的网络分割以及路由的重建.实验结果表明,这些算法和协议的引入节省了网络带宽,显著提高了系统的性能,保证了系统的自配置特性.

关键词: 发布/订阅;基于内容的路由;自配置;组播集群

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2003AA1Z2610 (国家高技术研究发展计划(863))

作者简介: **薛涛**(1973－),男,陕西西安人,博士生,讲师,主要研究领域为分布式计算,基于事件的中间件,分布式系统容错和可靠性算法;**冯博琴**(1942－),男,教授,博士生导师,主要研究领域为智能网络,编译技术.

中图法分类号: TP393 文献标识码: A

Internet 技术的广泛应用和移动、无线计算平台的快速发展,极大地改变了分布式系统的应用范围和规模.其带来的最大挑战之一是大规模的信息发布系统的广泛使用,例如股票与个性化新闻订阅系统、分布式拍卖系统、电子市场和电子商务应用等将会越来越普遍.传统的同步通信模型(CORBA,RMI,DCOM 等)由于其紧耦合的特点,难以适应这种大规模、异步和多点通信的需求.而发布/订阅通信模型由于具有异步、多点通信的特点,能够很好地满足 Internet 上大型应用系统松散通信的需要,它是构建大规模分布式系统的基石.这方面的研究受到了广泛的关注和重视.

发布/订阅系统包括信息的生产者(称为发布者)、信息的消费者(称为订阅者)以及事件代理.发布者发布信息(称为事件)给事件代理,订阅者向事件代理订阅感兴趣的事件,事件代理负责将接收的事件及时路由给感兴趣的订阅者.大部分商品化的发布/订阅系统都是基于主题的.在这种系统中,主题是一个字符串类型的关键字,它作为区分事件类型的标识,每个发布的事件都携带有主题信息;订阅者的订阅信息则包含某个主题,说明订阅者所感兴趣的主题.事件代理依据主题信息在事件和订阅信息中进行匹配,从而将事件转发给对它感兴趣的订阅者.基于主题的系统已经比较成熟,例如基于数据库的 SIFT 系统[1]、TIBCO 的 TIB/Rendezvous[2]、Talarian Corporation 的 SmartSocket[3]、IBM 的 MQSeries[4]以及规范的实现,包括 Java Message Service[5]和 CORBA Notification Service[6]等.

目前,基于内容的发布/订阅系统正在成为研究热点,实验的原型系统包括 Elvin[7],Gryphon[8,9],Siena[10,11]和 JEDI[12]等.这种系统中的事件不再依赖于外部的某个标准(如通道、主题等)分类,而是按照事件的内容本身分类.订阅者根据事件的内容来订阅事件,不必受系统预先定义的主题的限制.相比于基于主题的发布/订阅系统,基于内容的发布/订阅系统具有以下优点:订阅者可以自由地表达订阅信息,使系统更加灵活;由于不需要预先定义和维护大量的主题,使系统维护变得简单;此外,基于内容的发布/订阅系统更通用,它能轻易实现基于主题的发布/订阅系统,反之则不可能.

为了高效地实现基于内容的发布/订阅系统,有 3 个关键问题需要解决:(1) 事件代理采用何种算法能够实现事件和大量订阅者之间的高效匹配;(2) 考虑到系统必须运行在 Internet 这种广域网环境下,发布者、订阅者和事件的数量都很大,为此需要多个事件代理组成一个分布式的虚拟网络来提高系统的扩展性,产生的问题是事件如何在这个虚拟网络中进行高效的路由从而能够最终到达订阅者;(3) 虚拟网络应该是动态自适应的,即当节点或者链路失效后,网络应该自动重新组织、建立路由,而不需要人工干预.

本文主要就自配置策略和路由算法展开讨论.目前几乎所有的基于内容的发布订阅系统都忽略了系统的自配置特性,本文提出了解决该问题的模型和相关协议:将事件代理组织成具有组播集群(multicast clustering)结构的层次拓扑模型,通过组播集群复制协议和 CMTP(content-based multicast tree protocol)协议,对于节点或者链路失效导致的网络分割以及路由重建问题都能以简单、高效的算法处理.此外,在 Siena 和 JEDI 所采用的层次性路由算法基础上,我们设计了混合式路由算法,它借助组播集群充分利用物理网络的特性,节省带宽,提高了传输效率.

1 相关工作

有两种基本方法用于解决在基于内容的发布/订阅系统网络中路由的问题[9]:(1) 洪泛法,即事件在事件代理组成的虚拟网络中进行广播.每个事件代理接收到发布者或者相邻事件代理发送的事件,首先利用匹配算法得到相匹配的本地订阅者,将事件发送给这些订阅者.然后,将该事件转发给所有相邻事件代理(发送事件的事件代理除外).(2) 匹配优先法.每一个订阅者的订阅信息在虚拟网络中进行广播,这样,每个事件代理将事件和所有订阅信息匹配从而确定路由.由于不论是否有订阅者,事件都要发送给所有的事件代理,因此,洪泛法极大地浪费了网络带宽.匹配优先法适宜于小型系统,对于拥有成千上万的订阅者的大规模系统是不实际的.

Elvin 将事件代理组织成无环对等网络,它的路由算法没有公开,其最大的特点是淬火算法,该算法允许发

布者接收有关订阅者的信息,只发布订阅者订阅的事件,从而减少不必要的事件发布,节省网络带宽.Gryphon 将事件代理组织成层次性网络,它的路由算法采用的是匹配优先法,为了弥补系统性能的不足,匹配算法采用了高效的并行搜索树算法,同时研究了如何有效地利用组播以节省网络带宽.在大型系统中,订阅者订阅或者取消订阅导致的路由表的更新,代价很高,这限制了 Gryphon 的应用规模.Siena 实现了无环形和层次形两种虚拟网络,提出了合并(merging)、覆盖(covering)以及声明(advertisement)优化路由的算法.JEDI 用 Java 实现了类似于 Siena 的层次性网络,支持断开连接和重新连接操作,从而允许移动客户在事件代理之间进行迁移.Siena 和 JEDI 的层次性网络的路由算法和本文提出的路由算法相类似,不同的是,它们都没有考虑有关节点失效和节省网络带宽的问题,我们引入了组播集群加以解决.Elvin,Gryphon,Siena 和 JEDI 的事件代理虚拟网络都是静态配置的,不能进行网络的自我维护和管理.Host[13]是一个应用层的组播协议,它具有动态调整、自配置的特性,但是没有对基于内容的多点传输提供支持.

2 系统模型

图 1 描述了系统的层次性树状逻辑结构.下述各实体组成.

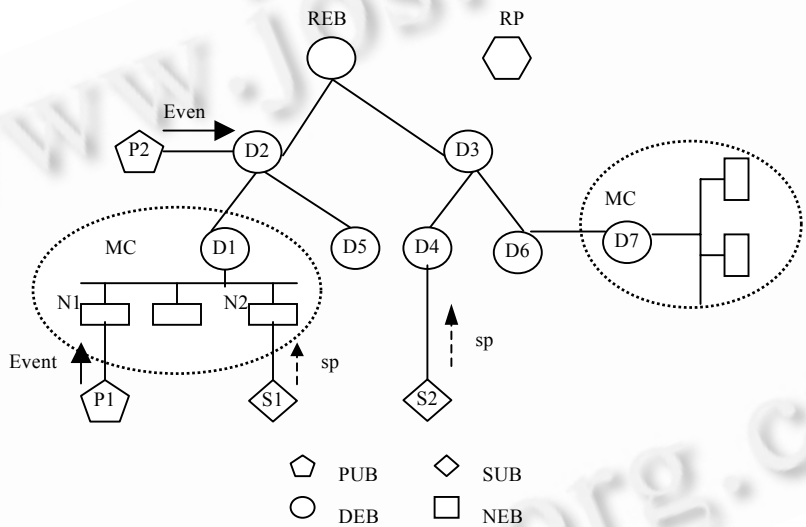


Fig.1 System architecture

图 1 系统体系结构

NEB(normal event broker):普通事件代理.它是处在一个支持组播的局域网(如以太网、令牌环网等)的事件代理,和本网内的其他 NEB 一起组成一个组播集群,能够利用组播进行相互通信.

DEB(designated event broker):指定事件代理,是构成层次性虚拟网络的节点.它是一个组播集群中负责与其他 DEB 相互通信的代表,一个组播集群只有一个 DEB.DEB 之间利用点对点单播通信.两个直接通信的 DEB 称为是相邻的.除根节点以外,每个 DEB 只有一个父节点,其他相邻节点称为孩子节点.DEB 的最大孩子节点数目称为度(degree).

REB(root event broker):根事件代理.它是一个特殊的 DEB,处于层次性虚拟网络最高层,没有父节点.

PUB(publisher):事件发布者.它与最近的事件代理(NEB 或者 DEB)建立网络连接,将事件发送给该事件代理,由事件代理将事件路由到各订阅者.

SUB(subscriber):事件订阅者.它与最近的事件代理(NEB 或者 DEB)建立网络连接,将订阅信息发送给该事件代理,事件代理依据订阅信息建立路由.

event:发布者发布的消息,称为事件.事件由一系列的名字-值对所组成的属性组成,我们定义描述 event 的模式为:(type,name,value),type 表示属性的类型,name 是字符串类型,表示属性名,value 表示该属性的值.

sp(subscription):订阅者发送的订阅信息,表示它所感兴趣的事件类别.由于我们以事件的内容作为分类,因此,sp 由一系列的对于事件属性的约束所组成,同一个 sp 的约束是“与”的关系.我们定义描述 sp 的模式为:(type, name,OP,value),其中 OP 是操作符,例如“<,>,=,between”等.一个 event 匹配一个 sp 当且仅当 sp 的所有约束被满足.

RP(rendezvous point):集中点.集中点是一个主机或者一个集群,它至少记录了当前层次性网络的根结点地址,以便提供给欲加入到网络中的新节点,从而能够使用 CMTP 协议找到最合适的父节点.RP 的地址是静态配置的.RP 不参与对事件的转发和路由,因此它的位置对系统没有影响.

MC(multicast clustering):组播集群,是一个支持 IP 多播的物理网络,如主机、以太网、Intranet 等.MC 中的事件代理都是 NEB,相互之间通过 IP 多播进行通信,其中一个 NEB 被选举作为 DEB,负责与其他 DEB 的通信.

3 基于内容的路由

洪泛法和优先匹配法都不适合于大规模广域网的系统应用,主要原因是这两种方法要求每一个 PUB 发布的 event 或者 SUB 发送的 sp 必须广播给所有的 DEB,每一个 DEB 要掌握全局信息才能决定合适的路由.如果 DEB 只根据相邻节点的信息就能决定路由,那么系统就会有很好的扩展性.基于这一思想,我们提出了一种混合式的路由算法.

```
[1]  Routing(message)
[2]  { if( message.type = event)
[3]    { if( this.type = NEB)
[4]        if( message.source.type = PUB)
[5]            multicast(message, MC)
[6]        else
[7]        {   ss := match(message)
[8]            for each connection in ss
[9]                send(message, connection)
[10]       }
[11]     else
[12]     {   ss := match(message)
[13]       for each connection in ss
[14]       { if( connection is MC)
[15]           if( message.source.type = NEB)
[16]               continue
[17]           else
[18]               multicast(message, MC)
[19]         else
[20]           send(message, connection)
[21]       }
[22]         if( this.type ≠ REB ∧ message.source ≠
       this.parent)
[23]             send(message, this.parent)
[24]     }
[25]   }
[26]   else if( message.type = sp)
[27]   {  if( this.type = NEB)
[28]     {  if( message.source.type = SUB)
[29]           multicast(message, MC)
[30]     }
[31]     else
[32]     {  addToRouteTable(sp, message.source)
[33]       if( this.type ≠ REB)
[34]           send(message, this.parent)
[35]     }
[36]   }
[37] }
```

Fig.2 Hybrid content-based routing algorithm

图 2 混合式路由算法

事件的传播分为两种情况:

(1) NEB 接收到 event.如果 event 由 PUB 发送,则在 MC 中,此 NEB 通过 IP 组播将 event 转发给它的 DEB;如果 event 由 DEB 或者其他 NEB 通过组播发送,则此 NEB 利用匹配算法将 event 与其路由表中的所有 sp 相匹配,得到匹配的 sp 连接列表,将 event 从相应的连接发送给订阅者.

(2) DEB 接收到 event.发送 event 的可能是 PUB、NEB、孩子节点或者父节点,无论是谁发送,首先利用匹配算法将 event 与其路由表中的所有 sp 相匹配,得到匹配的 sp 连接列表,将 event 从相应的连接发送,这里有一种情况除外:当 event 由 NEB 发送并且得到的匹配连接有指向 MC 的连接时则不向这些指向 MC 的连接发送

event,因为 MC 中的所有 NEB 已经通过组播接收到了此 event;第 2 步判断如果此 DEB 不是根节点并且发送 event 的不是父节点,则将 event 转发给父节点.

订阅的传播,同样分为两种情况:

(1) NEB 接收到 sp.如果 sp 由 SUB 发送,则在 MC 中,此 NEB 通过 IP 组播将 sp 转发给它的 DEB;如果 sp 由其他 NEB 通过组播发送,则不作任何处理.

(2) DEB 接收到 sp.发送 sp 的可能是 SUB、NEB 或者孩子节点,该 DEB 将 sp 以及和 sp 相对应的连接记录到路由表中,对于 NEB 发送的 sp 的相应连接指的是 MC 的组播地址(因为 IP 组播是无连接的);如果此 DEB 不是根节点,则将 sp 转发给父节点.

SUB 也可以取消订阅,对于取消的处理和对订阅的处理是类似的,在此不再赘述.

算法描述如图 2 所示.其中 match ()为匹配函数,将输入的事件和事件代理所记录的 sp 进行高速匹配,返回匹配成功的 sp 的相应连接.

我们以图 1 为例.订阅者 s1 向组播集群中的 N2 发出订阅 sp,N2 在 MC 中组播 sp(行 29),D1 将 sp 记入路由表中(行 32),然后向父节点 D2 转发 sp(行 34),D2 再发送 sp 至 REB(行 32～行 34).类似地,订阅者 s2 的订阅 sp 也分别发送至 D4,D3 和 REB(行 32～行 34).现在发布者 p1 发布事件给 N1,N1 在 MC 中组播 event(行 5),N2 和 D1 都接收到该事件.假设 event 和 sp 相匹配,则 N2 将 event 发送给 s1(行 7～行 9).此外,D1 将 event 发送给 D2,直至到达 REB(行 12～行 23).由于 event 和 sp 相匹配,REB 将 event 发送给 D3,D3 进行类似的动作,再发送给 D4,最后到达订阅者 s2(行 12～行 20).发布者 p2 也发布 event,到达 D2 后,由于和 s1 的 sp 匹配,D2 将 event 发送给 D1(行 12～行 20),并转发给父节点 REB(行 22～行 23).D1 将接收到的 event 在 MC 中组播(行 12～行 18),N2 接收到并发送至 s1(行 7～行 9).另外 event 也会逐步路由至 REB,D3,D4,到达 s2(行 12～行 20).

4 自配置策略

混合式路由算法不需要每个事件代理掌握全局信息,非常适合于在广域网环境下的应用,但是,由于层次性拓扑结构的特点,会产生任意一个节点或物理链路失效而导致的网络分割问题,因此,系统必须具有容错和自愈的能力.本文提出的组播集群复制协议和 CMTP 协议就分别解决了节点和物理链路失效的问题.

4.1 组播集群复制协议

为了防止由于 DEB 崩溃而导致的网络分割,使用组播集群对 DEB 进行容错复制,这样一旦 DEB 失效,可以由组播集群内的一个 NEB 代替失效的 DEB.整个过程分为两个阶段:选举阶段和探测恢复阶段.

4.1.1 选举阶段

每个 NEB 都有一个唯一的 id,在系统初始化时,NEB 将 id 连同它的 IP 向 DEB 注册,DEB 于是掌握集群内所有 NEB 的信息.DEB 周期性地组播“心跳”信息,声明它的存在.在“心跳”信息中,包括了所有 NEB 的 id 以及它的父节点和孩子节点的相关信息.此外,为了复制路由表,DEB 将每个建立的路由表项都在集群中组播,以便 NEB 建立同样的路由表.如果在一段时间内,NEB 没有接收到 DEB 的“心跳”,就认为 DEB 失效,于是启动选举算法.由于所有 NEB 已经通过 DEB 的“心跳”了解了其他 NEB 的 id,选定 id 最大的 NEB 为新的 DEB,并重新向新的 DEB 注册.id 最大的 NEB 将自己的类型设置为 DEB,接受其他 NEB 的注册,并且组播“心跳”信息.如果 id 最大的 NEB 也已经失效,那么其他 NEB 没有接收到“心跳”,就重新启动选举算法,选举 id 次之的 NEB 为 DEB,依此类推.

4.1.2 探测恢复阶段

每个 DEB 也周期性地向它的相邻节点(父节点和孩子节点)发送“心跳”信息,该“心跳”信息包含有集群内所有 NEB 的 id 和 IP 地址.这样,相邻 DEB 之间通过“心跳”交换各自集群内的 NEB 信息.如果一段时间内某个 DEB 没有“心跳”,则其相邻节点认为其发生了失效,启动探测算法.探测算法的目的是探测新的 DEB 以便建立新的连接.由于相邻节点已经通过“心跳”了解了失效节点的所有 NEB 信息,因此,首先选择失效节点的 id 最大的 NEB 为新的 DEB,然后向其发送探测消息,新的 DEB 接收到探测消息后,发出响应消息,相邻节点收到响应后,更新路

由表中的相应连接,这时就完成了系统的修复.如果未收到响应消息,则重试若干次,若还没有响应,则向 id 值次之的 NEB 发送探测消息.最坏情况下,若组播集群全部失效或者由于链路断开导致系统无法恢复,则必须启动 CMTP 协议,对虚拟网络进行重组.

4.2 CMTP协议

如果探测恢复阶段失败,则说明失效 DEB 的组播集群全部失效或者是物理链路断开,如果失效的是非叶子节点,则层次性的树形虚拟网络被分割,原来建立的路由也被破坏,导致部分事件不能到达订阅者.这时系统需要启动 CMTP 协议进行自愈处理.CMTP 协议需要解决两个问题:(1) 由于树形网络被分割,失效节点的孩子节点需要绕过失效的组播集群或者链路,重新加入到树形网络;(2) 要重新部署订阅信息 sp,以便和新形成的树形网络保持一致,重新建立路由表.

4.2.1 节点重新加入

为了减小树的深度以及尽量保持每个节点都具有最大的度,被分割的孩子节点需要尽可能地找到和它最近的节点作为父节点,这就需要在某个地方存储所有节点的相关信息列表,这限制了系统的扩展性,并且导致搜索效率降低以及管理维护的复杂性.CMTP 协议借鉴了 Host[13]的思想,采取了一种折中方法,它只从部分节点中查找最适宜的父节点,保证了系统的扩展性,尽力满足了系统的需要,但是不保证找到的父节点是最近的节点.

我们仍以图 1 为例.假设 D3 和 D4 之间的链路断开,D4 运行探测恢复阶段失败,则 D4 启动 CMTP 协议,重新加入网络.首先 D4 向集中点 RP 发出查询,RP 将保存的当前根节点 REB 返回给 D4,D4 然后向 REB 发出查询,REB 将自己的所有孩子节点返回给 D4,D4 通过某种测量措施从 REB 以及它的孩子节点中选择出最近的节点 D2,然后 D4 又向 D2 发出查询,如此重复这个过程,找到了最近的节点 D1,此时 D4 向 D1 发出加入请求.如果 D1 由于负载过重或者度已经到达最大,则拒绝该请求,D4 返回上一层,重新寻找,找到 D5,最终以 D5 作为父节点.

算法描述如图 3 所示.其中 stack 是一个堆栈,用来存放每一层的父节点及其相应的所有孩子节点;childrenList 是一个向量,用来存放某节点的所有孩子节点,向量的每一个分量都有一个 state 属性,如果没有搜索过,标记为 true,否则为 false;函数 findNearestNode()返回父节点和其孩子节点中 state 属性为 true 并且 TTL 最小的节点.

```
[1]  Join()
[2]  { root  := getRoot(RP)
[3]    parent := root
[4]    childrenList := getChildren(parent)
[5]    while(parent ≠ null)
[6]    {  nearestNode := findNearestNode(parent, childrenList)
[7]       if( nearestNode = null)
[8]       {  parent := stack.pop()
[9]          childrenList : = stack.pop()
[10]         continue
[11]      }
[12]      if( nearestNode ≠ parent)
[13]      {  stack.push(parent)
[14]         stack.push(childrenList)
[15]         parent := nearestNode
[16]         childrenList := getChildren(parent)
[17]         continue
[18]      }
[19]      else
[20]      {  response := sendJoinRequest(parent)
[21]         if ( response = false)
[22]         {  parent.state := false
[23]            continue
[24]         }
[25]         return parent
[26]      }
[27]   }
[28]  }
```

Fig.3 Join algorithm

图 3 节点加入算法

如果根结点 REB 失效,则其所有孩子节点在随机等待一段时间后向 RP 发出加入请求,第 1 个到达 RP 的孩子节点成为新的根节点.

4.2.2 路由的重构

树形网络拓扑结构的改变必然会导致对于事件路由的相应改变.例如,发布者 P2 发布的事件原来经过 D2,REB,D3,D4 到达订阅者 S2,当由于 D3 和 D4 之间的链路失效而使得 D4 重新加入并以 D5 作为父节点以后,P2 发布的事件将经过 D2,D5,D4 到达 S2,同时,事件将不再被转发给 D3.CMTP 协议要解决的第 2 个问题就是让所有受影响的节点重新建立路由,以便正确转发事件.路由的重构包括两个步骤:

(1) 去除老的路由.失效链路两端的节点将不再向对方发送对方感兴趣的事件,其效果就如同双方互相发出了取消订阅的消息.因此,去除老路由的方法就是两端节点自动产生对方取消订阅的消息,并按照取消订阅的算法处理.此时将删除路由表中的相应表项,并且向上层节点传播该取消订阅的消息.

(2) 建立新的路由.重新加入的节点将自己所有的订阅 sp 全部发送给新的父节点,父节点对这些 sp 依然按照订阅传播的算法来处理,从而重新建立了路由.例如,D3 和 D4 之间的链路失效,D3 产生对于 sp 的取消消息,并传播至 REB,从而去除了事件被转发至 D4 的路由;D4 重新加入网络后,将订阅 sp 传播给父节点 D5,D5 又将 sp 传播给 D2,此时新的路由就已经建立,由 P2 发布的事件将沿着 D2,D5,D4 到达 S2.

5 实 验

为了证明以上算法的可行性和正确性以及对其性能的评价,我们对算法进行了仿真实现,并且与没有组播集群的层次路由算法做了比较.仿真环境采用了 GT-ITM[14]模拟环境.

5.1 实验环境设置和评价指标

利用 GT-ITM 拓扑产生器的 Transit-Stub 图形模块生成树形网络,其中 DEB 节点 100 个,每个 DEB 的组播集群有 10 个 NEB,730 个链路,每个节点最大的度设置为 8.由于混合式路由算法相对于一般的层次性路由算法的改进在于增加了组播集群,从直观来讲,对于 SUB 的数量更为敏感,因此对于每一组实验,将 SUB 的数量设置为 X 轴的变量,其变化范围为 1～10000.在每一组实验中,PUB 的数量为常量,分为 100 和 1000 两类.为了简化实验,PUB 只发布一种事件,SUB 也只订阅这种事件,由于这种情况会对系统产生最大的负载,因此它的实验结果是可以类推到其他情况的.实验结果中的每个数据点都是以同样的参数运行了 10 次求得的平均值.

为了衡量系统的扩展性和效率,我们采用了两个评价指标:

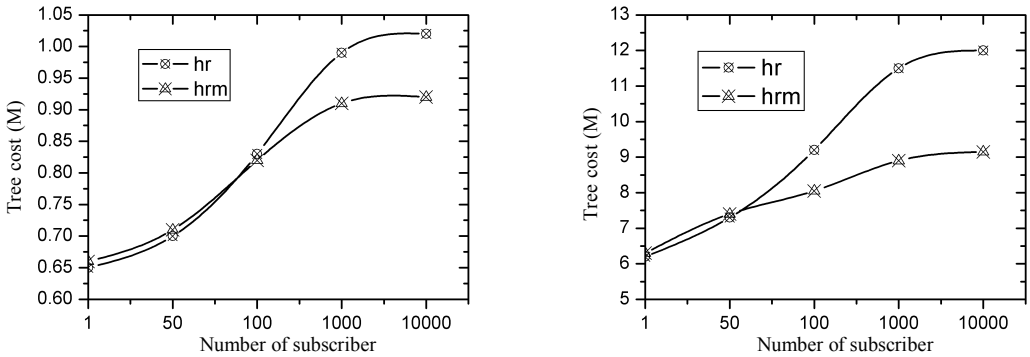
- 树的总代价 TCT(total cost of tree).TCT 是衡量发布订阅系统的一项基本指标,它是将所有链路的消息量进行求和得到的.由于网络带宽、延迟等的差别,TCT 的绝对值是没有意义的,但是通过相对比较,TCT 显示了系统扩展性的一个重要方面——系统负载的增加对于通信代价的影响.
- 订阅者接收事件的平均延迟时间 AED(average event delay),是所用时间和这段时间内接收到的事件数量的商,它指示了系统转发事件的效率.

图 4 和图 5 是实验结果,其中 hr 代表 JEDI,Siena 等采用的层次性路由算法,hrm 代表具有组播集群的混合式路由算法.

5.2 实验结果分析

(1) TCT:图 4 是层次路由算法和混合路由算法在 TCT 指标上的比较结果.从图中我们可以看到:1) 当订阅者数目超过 1000 时,两种算法的 TCT 都趋向一个恒定值,说明有一个临界点,当超过临界点后,增加订阅者并不会对系统增加额外的代价.产生的原因可能是超过临界点后,订阅者在每个节点上都形成聚集,因此,新增的订阅者不影响系统的负载总量.此外,临界点之后,混合路由算法比层次路由算法的 TCT 要小,而且发布者数量越大,二者的差距就越大.说明了混合式路由算法由于引入了组播集群,显著节省了带宽,提高了系统的扩展性.2) 在临界点之下,两种算法的 TCT 曲线都呈现近似线性增长,原因是订阅者逐步在节点中分布,在达到临界点之前,事件传播的代价也相应增加,直到到达临界点之后,代价才变为 0.图中有一个拐点,在拐点之前,两种算法的代价基本相同,拐点之后,混合路由算法比层次路由算法的 TCT 要小,并且随着订阅者数量的增加,差距也随之

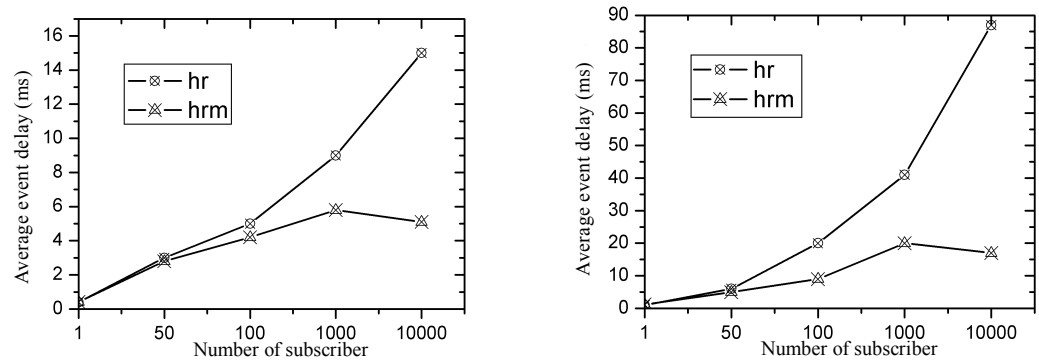
增大.发布者数量越大,该拐点就越提前.分析该现象的原因是,当订阅者数量较少时,在网络中较为分散,组播带来的效果并不明显;当订阅者数量增加时,其在各组播集群聚合的可能性增大,组播的优势就逐渐显现出来;同样,当发布者数量增加时,其与订阅者处在同一组播集群的可能性也增大了,因而利用组播节省了带宽.



(a) Publisher = 100 (b) Publisher = 1000

Fig.4 Comparison of TCT incurred by hierarchy routing algorithm and hybrid routing algorithm

图 4 层次路由算法和混合路由算法的 TCT 比较



(a) Publisher = 100 (b) Publisher = 1000

Fig.5 Comparison of AED incurred by hierarchy routing algorithm and hybrid routing algorithm

图 5 层次路由算法和混合路由算法的平均事件延迟比较

(2) AED:图 5 是层次路由算法和混合路由算法在平均延迟指标上的比较结果.从图中我们可以看到:1) 存在一个拐点,在拐点之前,两种算法的平均延迟几乎是相同的;在拐点之后,混合路由算法的平均延迟比层次路由算法的延迟小,并且随着订阅者数量的增加,其差距也在增大.拐点的位置受发布者数量的影响,发布者数量越大,拐点越提前,例如图中当发布者数量为 100 时,拐点的订阅者数量为 100,当发布者数量为 1000 时,拐点的订阅者数量为 50.显然该现象的原因也是由于随着订阅者和发布者数量的增加,利用组播的几率也相应增加,从而减小了延迟.2) 随着订阅者数量的增加,层次路由算法的平均事件延迟呈直线增长,并且越来越陡,系统性能严重恶化;但是对于混合路由算法,它的平均事件延迟在增长到一个临界点后,却呈下降趋势.产生的原因是超过临界点后,订阅者汇集在每个节点上的组播集群,事件的发送只需一次组播,因此事件到达新增的订阅者可以认为不需要花费额外的时间.可见,混合路由算法具有很强的扩展性,适宜于大型系统.

6 总 结

实现扩展性强的基于内容的发布/订阅系统,关键问题之一是事件代理组成的拓扑结构以及路由算法的设

计.目前大部分原型系统都采用的是层次性树形结构和层次性路由算法,但是没有利用物理网络组播的特性,本文在层次性树形网络的基础上引入了组播集群的结构,提出了混合式路由算法,实验结果表明,该算法在 TCT 和事件平均延迟两项重要指标比层次性路由算法有较大的改善,节省了网络带宽,显著提高了系统的扩展性和性能.

此外,目前所有原型系统的事件代理虚拟网络都是静态配置的,不具有自配置特性,任何一个节点或者链路的失效都会导致网络分割,系统较为脆弱.本文提出了组播集群复制协议和 CMTP 协议,对由于节点或者链路失效导致的网络分割以及路由重建问题都能以简单、高效的算法进行处理,当节点或者链路失效后,树形网络能够自动重新组织和建立路由而不需要人工干预,保证了系统具有很强的容错性.

References:

[1] Yan TW, Garcia-Molina H. The SIFT information dissemination system. ACM Trans. on Database Systems, 1999,24(4): 529−565.

[2] TIBCO. TIB/Rendezvous White Paper. http://www.tibco.com/software/enterprise_backbone/rendezvous.jsp

[3] Talarian Corporation. Everything you need to know about middleware: Mission-critical interprocess communication. White paper. Talarian Corporation, Los Altos, CA (now part of TIBCO, Palo Alto, CA), 1999. http://searchwebservices.techtarget.com/searchWebServices/downloads/Talarian.pdf

[4] IBM RedBook. Internet Application Development with MQSeries and Java. February 1997. IBM Corporation, Yorktown Heights, NY. http://publib-b.boulder.ibm.com/Redbooks.nsf/RedbookAbstracts/sg244896.html

[5] Sun Microsystems, Inc., Mountain View CA, U.S.A. Java Message Service, 1999. http://java.sun.com/products/jms/

[6] Object Management Group. Notification Service Specification, OMG Document Telecom/02-08-04. 2002. http://www.omg.org/docs/formal/02-08-04.pdf

[7] Segall B, Arnold D, Boot J, Henderson M, Phelps T. Content based routing with elvin4. In: Proc. of the Australian UNIX and Open Systems User Group Conference (AUUG2K). Canberra, Australian, Jun 2000. 25−30. http://elvin.dstc.edu.au/doc/papers/auug2k/auug2k.pdf

[8] IBM Corporation. Gryphon: Publish/subscribe over public networks. Technical report, IBM T. J. Watson Research Center, 2001. http://www.research.ibm.com/gryphon/papers/Gryphon-Overview.pdf

[9] Banavar G, Chandra T, Mukherjee B, Nagarajarao J, Strom RE, Sturman DC. An efficient multicast protocol for content-based publish-subscribe systems. In: Proc. of the IEEE Int'l Conf. on Distributed Computing Systems'99. New York: IEEE, 1999. 262−272.

[10] Carzaniga A, Rosenblum DS, Wolf AL. Design and evaluation of a wide-area event notification service. ACM Trans. on Computer Systems, 2001,19(3):332−383.

[11] Carzaniga A, Wolf AL. Content-Based networking: A new communication infrastructure. In: Makki SAM, ed. NSF Workshop on an Infrastructure for Mobile and Wireless Systems. LNCS 2538. Berlin: Springer-Verlag, 2002. 59−68.

[12] Cugola G, Nitto ED, Fuggetta A. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. IEEE Trans. on Software Engineering, 2001,27(9):827−850.

[13] Zhang B, Jamin S, Zhang L. Host multicast: A framework for delivering multicast to end users. In: Proc. of the IEEE INFOCOM 2002. Vol.3, New York: Institute of Electrical and Electronics Engineers Inc, 2002. 1366−1375.

[14] Zegura EW, Calvert K, Bhattacharjee S. How to model an internetwork. In: Proc. of the 1996 15th Annual Joint Conf. of the IEEE Computer and Communications Societies, INFOCOM'96. Vol.2, New York: Institute of Electrical and Electronics Engineers Inc., 1996. 594−602.