第34卷 第17期 计 算 机 工 程 2008年9月
Vol.34 No.17 Computer Engineering September 2008
·网络与通信· 文章编号：1000—3428(2008)17—0099—03 文献标识码：A 中图分类号：TP393

基于树状和分布式架构的MAP发现协议

王忠培[1]，周 健[2]
(1. 合肥工业大学计算机与信息学院，合肥 230009；2. 合肥工业大学网络与信息中心，合肥 230009)

摘 要：在分析原有HMIPv6网络架构的基础上提出一种基于树状和分布式架构相结合的网络架构。该架构可解决在HMIPv6中的MAP发现协议的不足，解决较高层次MAP的瓶颈问题和提高原有网络的鲁棒性。针对2种架构给出了各自的选择算法，论证该架构可适用于多种网络，有广泛的应用范围。

关键词：HMIPv6协议；树状结构；分布式结构；MAP发现

MAP Discovery Protocol Based on Integration of Tree and Distributed Framework

WANG Zhong-pei[1], ZHOU Jian[2]
(1. School of Computer & Information, Hefei University of Technology, Hefei 230009;
2. Center of Network & Information, Hefei University of Technology, Hefei 230009)

【Abstract】Based on performance analysis of the framework of HMIPv6, a new network framework combined with tree and distributed framework is proposed. The new network frame can solve the shortcoming of MAP discovery protocol in the HMIPv6, and the problems that high-level MAP is a bottleneck and improves the robust of mobile IP. Thus giving the selective arithmetic separately for the two framework, which demonstrates it can be suitable for various networks. So it can be used for wide range.

【Key words】HMIPv6 protocol; tree framework; distributed framework; MAP discovery

1 HMIPv6的原理及网络结构

针对移动IPv6(MIPv6)[1]的缺点，IETF制定了分级移动IPv6协议HMIPv6(Hierarchical Mobile IPv6)[2]，在HMIPv6网络架构中，引入一个新的实体——移动锚点MAP(Mobility Anchor Point)，它的作用相当于一个处于距MN较近的HA，MAP管理多个接入路由器AR(Access Router)，将它所管理的所有AR视为一个管理域，这样就把IPv6网络分成几个不同的域。其网络结构如图1所示。

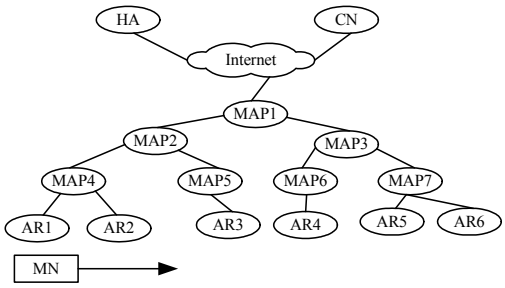

图1 树状结构的HMIPv6的网络结构

由图1可知，HMIPv6的架构是一个树形结构[3]，这种结构在具有减少通信流量、实现无缝移动等优点的同时，又具有以下缺点：

(1)较高层的MAP的瓶颈问题。如图1中的MAP2, MAP3和MAP1。因为较高层次的MAP数量一般较少，所以当有业务量通过时，没有多余的MAP去分担，大量的业务量就只能靠有限的MAP来分配，从而容易在这些点上造成拥塞，产生瓶颈问题。

(2)网络的鲁棒性比较差。当网络中的某一条链路断开，则下层的通信能力就会受到很大影响，甚至不能通信。如若这种断开是在更高层的链路上，所影响的范围将会更广。

2 HMIPv6中的MAP发现协议

针对树形结构的缺点，本文提出了一种基于树状和分布式架构[4]相结合的网络架构，如图2所示。

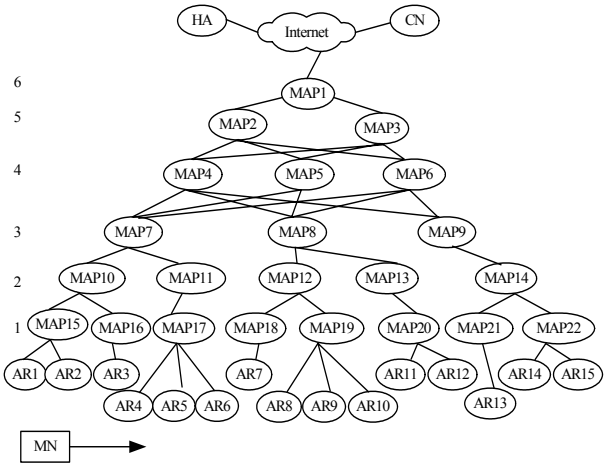

图2 基于树状和分布式架构相结合的网络结构

本文框架的提出是基于这样一种思路考虑：当MN的移

基金项目：安徽省教育厅自然科学基金资助项目(2005KJ049)
作者简介：王忠培(1981－)，男，硕士研究生，主研方向：下一代互联网，移动IPv6，无线网络；周 健，副教授
收稿日期：2007-10-10 **E-mail：**wzp004@sina.com

动仅仅是在一个较小范围内的微移动，此时的 MAP 选择是利用下面的树状结构。这种情况下的 MAP 选择采用的是最近共同祖先算法。在要使用这种算法之前，先要收集各个 MAP 所包含的 AR 信息以及各个 AR 所经由的各个 MAP。信息的收集采用下面的过程：在图 2 所示的树状结构中，每一个网络元素向上只有一个母体，向下可以有多个子体，这也正是树状结构的特点。为了让 AR 收集到它之上 MAP 信息，首先每个 AR 向上发送一条限定了跳数的信息，以表示自己的存在，在每经过一个 MAP 时跳数减 1，并且把它所经过的 MAP 添加到这个数据包中，这条消息一直向上传送，直到这条数据包的跳数减少到 0 或网络管理员限定的一个范围，这样的消息在每个 MAP 处进行汇总分析之后，每个 MAP 就都知道了自己域内的 AR 以及它们离自己的距离。以 MAP7 为例，经过上述过程的收集后，在它们的数据表中应该有这样的信息，如图 3 所示。

<table>
<tr><td rowspan="3">MAP7</td><td rowspan="2">MAP10</td><td>MAP15</td><td>AR1, AR2</td></tr>
<tr><td>MAP16</td><td>AR3</td></tr>
<tr><td>MAP11</td><td>MAP17</td><td>AR4, AR5, AR6</td></tr>
</table>

图 3　MAP7 所获得的网络拓扑信息

然后，每个 MAP 再按原路返回一条携带了这些等级拓扑信息的消息给它所管辖的 AR，这样，每个 AR 就知道了它之上的 MAP 所包含的 MAP 和 AR，以及它们离自己距离。最终，每个 AR 应具有这样一个数据库信息，以 AR1 为例，其所构造的表也同图 3。可以看出各个 MAP 所管辖的 MAP 及 AR，以及各个 AR 经过了哪些 MAP。比如从图 3 中，AR2 是先经过 MAP15，再经过 MAP10，最后到达了 MAP7。为了利用最近共同祖先算法，在当 MAP 按原路返回到 AR 时将它所经过的各个 MAP 存放在堆栈里。按照下面的最近共同祖先算法便可把任意 2 个 AR 之间的最近共同祖先算出来。

最近共同祖先算法：

```
Find_nearest_ancestor(SAR1,SAR2)
{Match=False;                    //判断是否找到了共同祖先
 InitStack(S);                   //初始化一个空的 S 栈
 While(not StackEmpty(SAR1))
 {
  A1=POP(SAR1);                  //对 SAR1 执行出栈操作
    While((not StackEmpty(SAR2))&&(Match==False))
        {
A2=POP(SAR2);
          PUSH(S,A2);            //将 A2 压入 S 栈
          If(A1==A2)
          Match=True;            //找到最近共同祖先
        }
 If(Match==False)
    While(not StackEmpty(S))
        {PUSH (SAR2,POP(S))   //恢复 SAR2 栈}
 Else
        Empty(SAR1);             //清空 SAR1 栈
 }
Return (S);   }
```

通过这种算法，可以得到 AR1 和 AR3 共同祖先是 MAP10，AR2 和 AR5 的共同祖先是 MAP7。

但针对本文提出的这种特殊结构，可以限定树状结构的层次数。这里初步设定为 3 层。如图 2 中的 1, 2, 3 数字所示。之所以定义 3 层，是基于这样的考虑：因为一个 MAP 域可管理多个 AR，这样最下层的 MAP 就已经管理了一定的范围。那么，其上的 MAP，如 MAP10, MAP7 所管理的范围更广。而据法国 INRIA 公司的研究表明，在通常的移动中，有 69% 的移动是在一个域内移动。因此，将其定义为 3 层，便能解决大部份的 MN 的微移动性问题。即使 MN 的移动是跨越多个 MAP 域，也可以通过利用树状结构采用最近共同祖先算法选择出 MAP。而且因为这里只定义了 3 层，所以即使最高层的 MAP(如 MAP7)与它所管理的 AR 之间的距离也不会很远，中间所经过的链路数较少，因此链路的断开几率便会很少。从而增加了网络的鲁棒性。即使某一条链路断开了，但只有 3 层，也可以很快的判断出究竟是哪条链路出故障，从而很快的找到问题所在，加以解决。

可见，树状结构主要解决了 MN 的微移动性问题，而当 MN 的移动性较强，如车载设备，在较短的时间内可能跨过多个 MAP 区域，如车载设备从 AR1 移动到 AR13，此时的 MAP 选择便利用了图 2 所示的上层分布式结构模型。随着今后的发展，移动终端的大范围的移动增多，从而这种较大范围的宏移动会经常发生。若采用的是单一的树状结构，由于上层 MAP 的数量相对于下层 MAP 的数量较少，从而 MAP 瓶颈问题将会变得更加突出。更坏的情况是，上层某条链路断开，如图 2 所示 MAP1 和 MAP2 之间的链路断开了，若仅采用了树状结构，则 MAP2 所管理范围将会发生通信中断，这样便会造成较大范围的通信中断，所产生的影响可想而知，而且由于链路数较多，要想快速地找出断开的链路，也会变得并不容易。但若采用了分布式结构，即使某条链路断开，它还可以依据一定的算法选择出另一条链路继续通信，而即使选择的另一条链路又断开了，它还可以再选择其他链路。可见，这种方式的鲁棒性是很高的。另外，这种架构的另一个优点就是还可以解决高层 MAP 的瓶颈问题。因为通信量并不是仅仅在几个较高层 MAP 里通过，而是动态选择 MAP 路径，所以业务量的通过所经过的 MAP 是动态变化的，从而不会产生瓶颈问题。这里设计了一个动态的路径选择算法。根据图论理论，设计了求出任意两顶点间的最短路径问题的算法[5]。

求任意两顶点的最短路问题采用矩阵求解法，其主要思想如下：

设图 G 的顶点集为 $V=\{v_1, v_2, \cdots, v_n\}$，记 $V_1=\varnothing$，$V_2=\{v_1\}, \cdots, V_n=\{v_1, v_2, \cdots, V_{n-1}\}$，$V_{n+1}=\{v_1, v_2, \cdots, v_n\}$。

称矩阵 $\boldsymbol{L}=(l_{ij})_{n\times n}$ 为图 G 的长度矩阵：

$$\boldsymbol{L}=\begin{pmatrix} l_{11} & l_{12} & \cdots & l_{1n} \\ \vdots & & & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{pmatrix}$$

定义 $d_{ij}^{(m)}$ 为 G 中从顶点 v_i 到顶点 v_j 的内部顶点(路径中内部点)，只能取自 V_m，且 V_{m-1} 最多只能取一次的最短路径的长。

显然，对于一切 $r<m$，$d_{ij}^{(r)}$ 表示从顶点 v_i 到顶点 v_j 的内部顶点都属于 V_r 的最短路径的长，$d_{ii}^{(r)}=0$，由此不难得到关于 $d_{ij}^{(m)}$ 的递推公式：$d_{ij}^{(0)}=l_{ij}$ (即边 v_i v_j 的权)，$d_{ij}^{(m)}=\min\{d_{ij}^{(m-1)}, d_{im}^{(m-1)}+d_{mj}^{(m-1)}\}$, $i\neq j$。

矩阵算法：

从 G 的长度矩阵 $\boldsymbol{D}^{(0)}=(l_{ij})$ 开始，依次构造出 n 个矩阵 $\boldsymbol{D}^{(1)}, \boldsymbol{D}^{(2)}, \cdots, \boldsymbol{D}^{(n)}$。若 $\boldsymbol{D}^{(m-1)}=(d_{ij}^{(m-1)})$，则第 m 个矩阵 $\boldsymbol{D}^{(m)}=$

($d_{ij}^{(m)}$)定义为：$d_{ij}^{(m)}=\min\{d_{ij}^{(m-1)},\ d_{im}^{(m-1)}+d_{mj}^{(m-1)}\}$, $i\neq j$。

运算过程从 $m=1$ 开始，让 i, j 分别取遍 1 到 n 的所有值，然后，m 增加 1，反复进行，直到 $m=n$ 时停止，这时 $\boldsymbol{D}^{(n)}=(d_{ij}^{(n)})$中元素 $d_{ij}^{(n)}$ 就是从顶点 v_i 到 v_j 的最短路径长。设有如图 4 所示的赋权图，求任意两顶点的最短路长。

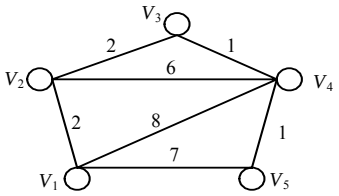

图 4　初始赋权图

由图 4 可构造出初始矩阵 $\boldsymbol{D}^{(0)}$ 为

$$\boldsymbol{D}^{(0)}=\left(l_{ij}\right)=v_1\begin{pmatrix}0&2&\infty&8&7\\2&0&2&6&\infty\\\infty&2&0&1&\infty\\8&6&1&0&1\\7&\infty&\infty&1&0\end{pmatrix}$$

由 $\boldsymbol{D}^{(0)}$ 再求 $\boldsymbol{D}^{(1)}=(d_{ij}^{(1)})$：$d_{11}^{(1)}=0$；$d_{12}^{(1)}=\min\{d_{12}^{(0)},\ d_{11}^{(0)}+d_{12}^{(0)}\}=\min\{2,0+2\}=2$; $d_{13}^{(1)}=\min\{d_{13}^{(0)},\ d_{11}^{(0)}+d_{13}^{(0)}\}=d_{13}^{(0)}=\infty$；…，$d_{25}^{(1)}=\min\{d_{25}^{(0)},\ d_{21}^{(0)}+d_{15}^{(0)}\}=\min\{\infty, 2+7\}=9$; $d_{24}^{(1)}=\min\{d_{24}^{(0)}, d_{21}^{(0)}+d_{14}^{(0)}\}=\min\{6,2+8\}=6$;…

于是：

$$\boldsymbol{D}^{(1)}=\begin{pmatrix}0&2&\infty&8&7\\2&0&2&6&9\\\infty&2&0&1&\infty\\8&6&1&0&1\\7&9&\infty&1&0\end{pmatrix}$$

同样由 $\boldsymbol{D}^{(1)}$ 求 $\boldsymbol{D}^{(2)}$：

$$\boldsymbol{D}^{(2)}=\begin{pmatrix}0&2&4&8&7\\2&0&2&6&9\\4&2&0&1&11\\8&6&1&0&1\\7&9&11&1&0\end{pmatrix}$$

同理

$$\boldsymbol{D}^{(3)}=\begin{pmatrix}0&2&4&5&7\\2&0&2&3&9\\4&2&0&1&11\\5&3&1&0&1\\7&9&11&1&0\end{pmatrix}$$

$$\boldsymbol{D}^{(4)}=\boldsymbol{D}^{(5)}=\begin{pmatrix}0&2&4&5&6\\2&0&2&3&4\\4&2&0&1&2\\5&3&1&0&1\\6&4&2&1&0\end{pmatrix}$$

上述算法是基于一开始各边的权值是已知的情况下所进行的选择算法，而在实际应用中，开始并不知道各边的权值是多少，为了得到初始权值，本文采用依层次结构给出各条链路的一个初始值，但这个初始权值并不是随意给出，是根据层次结构给出的。考虑到 MAP 选择时，先是选择较低层次的 MAP，所以较低层的 MAP 给出一个较低的权值，而较高层的给出一个较大的权值。如图 2 所示，在 3~4 层之间的连线给出的初始权值为 1，4~5 层之间的连线给出的初始权值为 2，5 层~6 层之间的连线给出的初始权值为 3，若有更多层，则每升高一层，数值加 1。比如 MAP4 和 MAP7 的初始值为 1，MAP2 和 MAP5 的初始值为 2，MAP1 和 MAP3 的初始值为 3，其他连线按此方式同样给出。而动态路径选择的权值是基于这样一种思想：即当某次选择了一条最短路径后，便把沿途所经过的 MAP 之间的路径的权值加 1。比如若选择了 MAP7－MAP4－MAP2－MAP5－MAP8 的这样一条路径，便把 MAP7 和 MAP4、MAP4 和 MAP2、MAP2 和 MAP5、MAP5 和 MAP8 的权值加 1。此时 MAP7 和 MAP4 之间的权值变为 4，MAP4 和 MAP2 之间的权值变为 3，MAP2 和 MAP5 之间的也变为 3，MAP5 和 MAP8 之间的变为 4。这样构造出一个新的带权的图。后面的最短路径选择再以此时所构造出的带权图为基础，进行新一轮的选择。

另外，从算法中还可以看到，矩阵大小是由图的顶点数决定的，为了在计算机中能更快地计算，可以适当地控制 MAP 的个数，图 2 给出了 3 层的结构，包含了 9 个结点，虽然给出的顶点数较多，增加了计算的时间，但图中考虑的是在一个较大范围内的 MN 的宏移动问题，它所覆盖的范围很大，而在实际应用中，可以根据当时的情况，适当减少 MAP 的个数而减少计算机的处理时间。其次，当在较大覆盖范围里时，也可以减少其中的一些不必要的 MAP 点，比如在图 2 中，MAP5 便可以给精简掉。从而减少了计算机的计算时间。

3　本文架构在实际中的适用情况

(1)若 MN 总是在域内移动或跨域的个数不多，就可以只使用单一的树状结构即可。比如 MN 分别在 AR1, AR2 之间，AR4, AR5, AR6 之间，AR2, AR3, AR4, AR5 之间移动时。而唯一不同的是，树状结构根据实际情况会由几层构成。

(2)若 MN 大部分是在做较大范围的跨越多层移动时，如车载系统，便采用两种结构相结合的架构。但由于是较大范围的宏移动，因此可以适当减少树状结构的层次，而增加分布式结构的层次。

(3)若 MN 移动性介于上面讨论的两者之间，则采用 2 种架构相结合的方式，但层次数相对于上面(2)中讨论的大范围的宏移动的情况，层次数应该减少。比如 2 种结构都是 2 层或者都是 3 层。具体是 2 层还是 3 层，可根据 MAP 所覆盖的范围以及实际物理位置的范围。若 MAP 域的管辖范围较大，可采用 2 层；若较小，可采用 3 层。而当 MN 的经常是大范围的物理移动，可采用 3 层；而若物理移动范围较小，采用 2 层即可。

综上，具体采用哪种架构以及 2 种架构里使用几层，需根据 MAP 所能覆盖的范围大小以及当时 MN 的物理范围的移动性而定。具体情况仍需具体分析。

4　有待进一步考虑的问题

在分布式结构的最短路径选择算法中，所赋的权值大小仅仅根据路径选择时所经过的次数。这种考虑还过于单一，有可能选择出的并不是最优路径。因此，在今后应将多种因素进行综合考虑，给出一个较优的权值。如，可以再综合考虑通信链路之间的带宽大小作为权值大小确定的依据之一。

5　结束语

在对原有 HMIPv6 分析的基础上，指出了原有网络架构中的不足之处。为了克服这些缺点，本文提出的方案可以在

（下转第 126 页）