支持压缩和多下一跳查找的路由查找方案*

梁志勇, 徐 恪+, 吴建平, 徐明伟

(清华大学 计算机科学与技术系,北京 100084)

An IP Lookup Scheme Supporting Routing Compaction and Multi Next Hops

LIANG Zhi-Yong, XU Ke+, WU Jian-Ping, XU Ming-Wei

(Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

+ Corresponding author: Phn: +86-10-62785822, E-mail: xuke@csnet1.cs.tsinghua.edu.cn, http://netlab.cs.tsinghua.edu.cn

Received 2003-04-01; Accepted 2003-08-12

Liang ZY, Xu K, Wu JP, Xu MW. An IP lookup scheme supporting routing compaction and multi next hops. *Journal of Software*, 2004,15(4):550~560.

http://www.jos.org.cn/1000-9825/15/550.htm

Abstract: TCAM (ternary content addressable memory) is a popular device for fast routing lookup. The advantages of TCAM are high lookup speed and simple operation. However, TCAM still has three explicit disadvantages: high cost, high power consumption and complex update. For load balancing and policy routing, routers have to hold considerable routes with multi next hops. This paper proposes a fast IP lookup scheme that is based on TCAM and supports routing lookup for multi next hops. With two index tables, the scheme can store and retrieve the routes with multi next hops. To improve the TCAM update, a fast update algorithm—*N*-subspace algorithm that can approximately reach the *O*(1) complexity for current Internet routing tables is also proposed. To decrease cost and power consumption, an efficient routing compaction method is also applied, which is based on the Trie structure and can reduce 20% routes for Internet routing tables. Also, the scheme can scale to IPv6 easily.

Key words: routing lookup; routing update; routing compaction; multi next hops; TCAM (ternary content addressable memory)

摘 要: TCAM(ternary content addressable memory)是目前流行的一种高速路由查找技术.TCAM 具有查找速度快、操作简单的优点,但同时它也具有 3 个明显的缺点:成本高、功耗大和路由更新复杂.路由器为了实现负载平衡以及策略路由,在路由表中保存着相当数量的具有多个下一跳的路由表项.基于 TCAM 技术,提出一种支持多下一跳的高速路由查找方案.方案通过两级索引表实现了多下一跳路由的存储和快速访问.为了提高

* Supported by the National Natural Science Foundation of China under Grant No.90104002 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant Nos.2001AA121013, 2001AA112132 (国家高技术研究发展计划(863)); the Key Science and Technology Project of Ministry of Education of China under Grant No.02004 (教育部科学技术重点项目)

作者简介: 梁志勇(1978－),男,广西容县人,硕士,主要研究领域为路由查找算法;徐恪(1974－),男,博士,讲师,主要研究领域为计算机网络体系结构,高性能路由器体系结构,分布式实时操作系统;吴建平(1953－),男,博士,教授,博士生导师,主要研究领域为计算机网络体系结构,计算机网络协议测试,形式化技术;徐明伟(1971－),男,博士,副教授,主要研究领域为计算机网络体系结构,形式化方法,协议一致性测试.

TCAM 的更新效率,方案还提出了一个 N 子空间 TCAM 更新算法.该算法对目前实际网络中的路由表,可达到近似 $O(1)$的更新复杂度.为了减少 TCAM 的成本和功耗,方案中还使用了有效的路由压缩技术.压缩技术基于 Trie 树结构,实现简单.应用压缩技术,对于实际网络中的路由表,可减少 20%的路由.该查找方案可以很容易地应用到未来的 IPv6 网络中.

关键词: 路由查找;路由更新;路由压缩;多下一跳;TCAM

中图法分类号: TP393 **文献标识码**: A

路由器是 Internet 网络中的重要设备,它负责对网络中的分组进行路由转发.转发分组时,路由器需要基于每个分组头部的目的 IP 地址,在路由表中进行路由查找,找到分组正确的下一跳出口.由于路由器为每个分组都进行路由查找操作,路由查找的性能直接影响着路由器整体性能.20 世纪 90 年代,IETF 为了解决地址浪费以及路由表规模增长过快等问题,提出 CIDR(classless inter-domain routing)地址结构[1].CIDR 允许任意长度的网络地址前缀存在,路由查找过程变为最长匹配查找过程,这极大地增加了路由查找的复杂性.下一代 Internet 采用 IPv6 技术[2],地址长度从 32 位扩展为 128 位,路由表规模进一步增加,路由查找难度也进一步提高.目前,为了满足网络带宽的需求,路由器不得不采用高速率的接口.OC48 和 OC192 等速率的接口已经在路由器中得到应用.在 OC192 速率下,路由器每秒钟需要转发 3 000 万个分组,每秒钟需要进行 3 000 万次的路由查找.高速的路由查找技术成为高性能路由器设计的关键.

为了解决高速路由查找问题,研究人员提出了多种不同的解决方案[3~7].目前,TCAM 是一种比较流行的硬件查找技术,它可以实现高速的路由查找.TCAM 芯片内部采用并行技术,可获得 $O(1)$的查找复杂度.目前市场上的 TCAM 芯片的查找速度最快可达 100M 次/s.与其他路由查找技术相比,TCAM 的技术优势在于查找速度快,实现简单,但它也存在下面 3 个缺点:(1) 成本高.TCAM 芯片要比相同存储空间的 SRAM,DRAM 贵很多.(2) 功耗大.TCAM 芯片内部采用并行技术进行关键字的比较,内部的功耗很大.(3) 路由更新复杂.在用 TCAM 技术实现最长前缀查找时,路由前缀在 TCAM 芯片内需要按照一定的顺序进行排序,这使得路由更新操作相对复杂.低效率的路由更新会极大地影响路由查找性能.

针对 TCAM 技术的 3 个缺点,研究人员提出了多种解决方案.对于缺点(1),(2),可以通过压缩路由表的方法来解决.压缩后的路由表对 TCAM 存储空间的需求减少,使用较小容量的 TCAM 芯片就可以满足存储要求,这就降低了 TCAM 技术的成本,同时使用小容量的 TCAM 芯片也减少了芯片内部的功耗以及散热.Liu 基于 Prunning 和 Mask Extension 技术,提出一种 TCAM 路由表的压缩算法[8].该算法对 Internet 中实际路由表可进行有效的压缩.但算法实现复杂,更新速度很慢,难以满足 Internet 核心路由器路由更新的速度.

对于缺点(3),Shah 提出两种有效的 TCAM 路由更新算法[9].其中 CAO_OPT 算法,可达到 $O(D/2)$(D 为 Trie 树中最大链长度)的更新复杂度,平均情况下,CAO_OPT 算法的更新复杂度可达 $O(AD/2)$(AD 为 Trie 树中所有链的平均长度).当路由表中路由前缀比较少时,AD 值也比较小,CAO_OPT 算法有着很好的性能.但随着路由前缀的增加,Trie 树结点分布越来越密集,AD 值也随之增加,CAO_OPT 算法性能相应下降.尤其对 IPv6 协议来讲,地址长度从 32Bits 扩展为 128Bits,路由前缀数目大大增加,CAO_OPT 算法的性能会下降很多.

路由表是路由器中重要的数据单元,它是路由查找转发的依据.由于安全、费用和带宽等因素,路由器需要为某些数据流做负载平衡和策略路由,在路由表中为一些目的网络保存多个下一跳信息,因此路由器的路由表中存在着相当数量的多下一跳路由.多下一跳路由的存在是路由表的重要特征之一,它增加了路由查找方案设计的复杂度.目前还没有研究者提出支持多下一跳路由的高速路由查找方案.

本文基于 TCAM 技术,提出一种高速的路由查找方案.该方案使用两级索引结构支持多下一跳路由的查找,可提供 100M 次/s 的查找速度.为了改善 TCAM 技术的 3 个缺点,本文提出一个 TCAM 路由更新算法.最好情况下,该算法具有 $O(D/2\times N)$的更新复杂度(D 为 Trie 树中最大链长度,N 为大于等于 2 的整数);平均情况下,算法更新性能也大大优于 CAO_OPT 算法[9].对于实际 Internet 中的路由表,该算法可以达到近似 $O(1)$的更新复杂度.在保证更新速度的前提下,该方案还应用一个有效的路由压缩技术对路由表进行压缩,从而降低 TCAM 的成本和功耗.该方案设计考虑了对 IPv6 协议的扩展性,可以很容易地从 IPv4 升级到 IPv6.

1 Internet 路由表特点

为了进行方案设计,我们对 Internet 上实际的路由表[10]进行了考察.考察的方面主要包括:路由表中多下一跳路由的分布情况和相关的一些信息.

表 1 是对 5 个路由表下一跳信息的统计.我们考查的 5 个路由表都为 BGP 路由表,并不是路由器内用于转发的真实路由表,Internet 中比较大规模的转发路由表是很难获得的,但 BGP 路由是路由最主要的来源,BGP 路由表可以近似地反映出真实路由表的情况.从表 1 中我们可以总结出 Internet 路由表的下面几个特点:

(1) Internet 路由表中存在着相当数量的多下一跳路由.5 个路由表中,Mae-West 的多下一跳路由所占比例最大,接近 50%;Aads 的多下一跳路由所占比例最小,但也达到了 19%.路由器内用于转发的真实路由表多下一跳路由所占的比例可能达不到表 1 中的比例,但多下一跳路由在路由表中的存在是不能忽视的,在进行路由查找算法设计时,多下一跳路由的查找应该被考虑进来.

(2) 绝大部分多下一跳路由的下一跳个数都不大于 4.在 5 个路由表中,具有 4 个以上下一跳的路由所占路由表的比例都是很小的.其中 Aads 和 Pacific Bell 最小,为 0;Mae-West 最大,为 2.61%.

(3) 单条路由具有的最多的下一跳个数小于 8.Mae-West 路由表中的路由具有最多的下一跳数 7.

(4) 路由表中不同的下一跳的总数是比较少的.从表 1 中我们可以清楚地看到这一点.在 5 个路由表中,所有路由具有的不同下一跳数目最多的也未超过 35.路由表具有这样的特点,主要是因为路由器的接口数相对较少,这使得路由器连接的下一跳路由器的个数也比较少,反映在路由表中就是不同的下一跳数目比较少.

Table 1 Next hop statistics for five routing tables

表 1 5 种路由表下一跳信息统计

Routing table	Mae-East	Mae-West	Aads	Paix	Pacific Bell
Route number (K)	22	34	30	16	42
Routes with multi next hops (%)	37.03	49.02	19.44	43.96	26.94
Routes with two next hops (%)	31.58	33.48	16.39	34.18	21.3
Routes with three next hops (%)	4.39	9.07	2.72	8.09	4.89
Routes with four next hops (%)	1.02	3.85	0.33	1.08	0.75
Routes with more than 4 next hops (%)	0.05	2.61	0	0.61	0
Most next hops of a route	5	7	4	6	4
Different next hops	30	33	26	20	2

上述 Internet 路由表的 4 个特点是我们方案设计的基础,其中,后 3 个特点在方案中得到充分利用,是方案设计的重要依据.

2 方案总体设计

2.1 总体框架

图 1 为方案的总体设计.图 1 包括 5 个部分:TCAM 芯片、一级下一跳索引表、二级下一跳索引表、下一跳映射表以及 FPGA(field programmable gate array)控制模块.

TCAM 芯片存储路由前缀信息,即(前缀 IP,前缀掩码).TCAM 完成路由前缀的最长匹配查找.

一级索引表存储单元与 TCAM 芯片的存储单元一一对应.比如图 1 中前缀为 1.0.0.0/8 的路由,它的前缀信息存储在 TCAM 芯片中的第 k 个单元,则相应的一级索引信息存储在一级索引表的第 k 个单元.一级索引表的存储单元宽度为 16 bits(这个宽度可根据二级索引表的大小进行调整),它有如图 2 所示的 3 类结构.图 2(a)单元表示该路由表项只存在一个下一跳,单元后 15 bits 存储跳映射表的索引,根据这个索引,可以在映射表中得到(下一跳 IP,出端口)的转发信息.图 2(b)单元表示该路由表项存在 2,3 或 4 个下一跳.Block number(BN)为二级下一跳索引表的块编号,它的取值可为 0,1,2,分别对应下一跳个数为 2,3,4 的路由.Block offset(BO)为二级索引表的块内偏移.图 2(c)单元表示该路由表项存在两个或两个以上的下一跳.BN 的取值为 3,next hop number(NHN)为路由表项下一跳的个数,它的取值可为 2~8(上限为 8 是因为第 1 节提到的路由具有最大的下一跳数小于 8),BO 为二级索引表的块内偏移.BN,BO 和 NHN 可用来访问二级索引表.

二级索引表存储映射表的索引,单元数据宽度为 8 bits(后面会解释选择 8bits 的原因).二级索引表分为 4 个

块.块 1、块 2、块 3 分别存储下一跳个数为 2,3,4 的路由表项的映射表索引.如图 1 所示,块 2 的灰色区域就存储了某个路由的两个下一跳的映射表索引.块 4 可存储下一跳个数为 2~8 之间的路由表项的映射表索引.块 4 一般不存储下一跳个数为 2,3,4 的路由表项.只有在块 1、块 2 和块 3 无空余空间时,才会存储相应的路由表项.路由表项多个下一跳的映射表索引在二级索引表中连续存储.比如,某个路由表项有 k 个下一跳,第 1 个下一跳的映射表索引存储在二级索引表的第 n 个单元,则其余 k–1 个映射表索引依次存储在 n+1,n+2,…,n+k–1 的 k–1 个连续单元里.

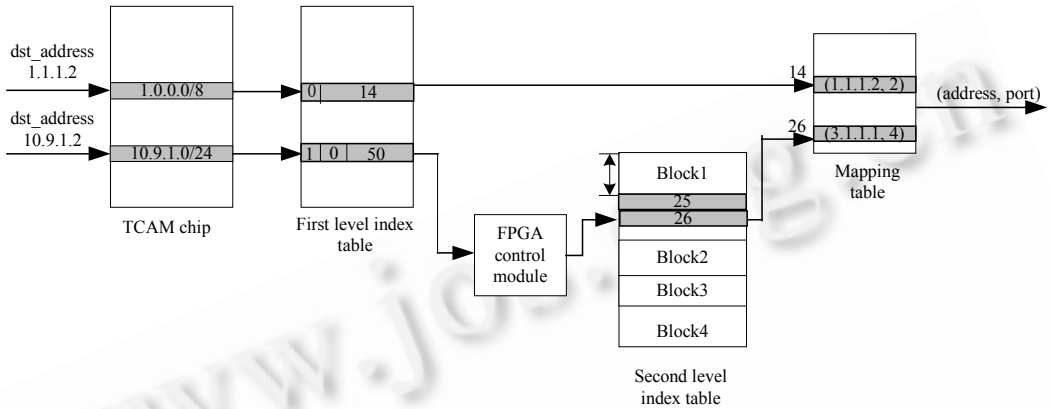

Fig.1 Scheme framework

图 1 方案框架

二级索引表是实现多下一跳查找支持的关键部分.二级索引表分为 4 个块,这主要利用第 1 节提到的 Internet 路由表中绝大多数路由的下一跳个数都小于等于 4 的特点.块 1、块 2 和块 3 可满足绝大多数路由存储需要,块 4 满足特殊路由的存储.二级索引表中块 1、块 2、块 3 的大小可根据实际路由表中多下一跳路由表项的统计规律来确定,比如下一跳个数为 2,3,4 的路由表项所占百分比为 r_2%,r_3%,r_4%,一级索引表的深度为 N,则块 1、块 2 和块 3 的大小分别为 2×N×r_2%,3×N×r_3%,4×N×r_4%.块 4 的大小可根据实际情况来决定.

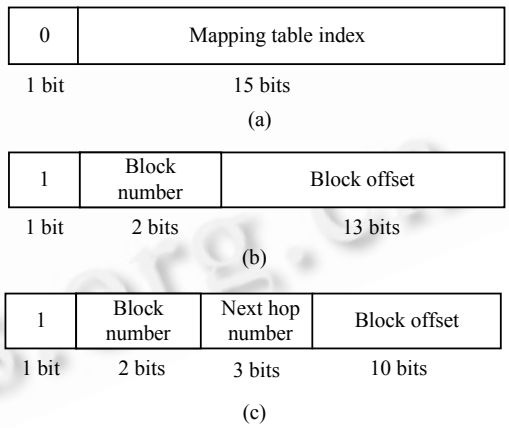

Fig.2 Unit structure of the first level index table

图 2 一级下一跳索引表单元结构

跳映射表单元存储转发信息——下一跳 IP,出端口.由于路由表中所有路由具有的不同下一跳的数目比较少,所以映射表的深度选为 256.深度选为 256 可以有效地减少二级索引表的存储空间,由于映射表的深度为 256,二级索引表存储着映射表的索引,所以 8 bits 的数据宽度就可以满足二级索引表的设计要求.

FPGA 控制模块完成路由查找过程中所有的逻辑计算,同时还存储着二级索引表 4 个块的基地址(base address,简称 BA).FPGA 控制模块的一个重要功能是通过一级索引表单元中的 BN,BO 和 NHN 域计算出要访问的二级索引表单元位置,FPGA 控制模块的具体计算过程如下:

(1) 通过 BN,得到相应块的基地址 BA;

(2) 如果 BN 小于 3,则通过 BN 得到对应的 NHN;具体来说,BN=0,1,2 分别对应 NHN=2,3,4;如果 BN=4,NHN 可从一级索引表单元中得到;

(3) 根据转发策略,从多个下一跳中选择一个作为转发的下一跳;简单的策略是,可以根据(分组的源地址+分组的目的地址)/NHN 得到选择的下一跳的索引(index);

(4) 最后计算获得要访问的二级索引表的单元位置(BA+BO+ index).

2.2 路由查找过程

整个查找过程可用下面的伪代码来描述.

```
Lookup()
{
    查找 TCAM 芯片得到前缀位置 k;
    访问一级索引表单元 k;
    if (单元 k 第 1 个 bit 为 0)
    {
        根据单元 k 中映射表索引访问映射表;
        得到下一跳 IP 地址和出端口;
    }else
    {
        访问单元 k 得到 BN,BO 和 NHN;
        把 BN,BO 和 NHN 交给 FPGA 控制模块;
        通过 FPGA 计算得到块基地址 BA 和选择下一跳的索引 index;
        访问二级索引表中(BA+BO+index)单元;
        根据二级索引表单元中的映射表索引访问映射表;
        得到下一跳 IP 地址和出端口;
    }
}
```

图 1 中包括两个具体路由查找的例子.例 1 为对目的地址 1.1.1.2 的查找.首先在 TCAM 芯片内根据最长匹配原则,找到与之匹配的前缀 1.0.0.0/8.根据前缀 1.0.0.0/8 所在单元的位置,访问对应的一级索引表单元.一级索引表单元的内容为(0,14),单元第 1 个 bit 位为 0,所以单元后 15 个 bits 的内容为映射表的索引(14).访问索引表的第 14 个单元得到(下一跳 IP,出端口)的转发信息(1.1.1.2,2).例 2 为对目的地址 10.9.1.2 的查找.首先在 TCAM 芯片内找到与之最长匹配的前缀 10.9.1.0/24.根据前缀 10.9.1.0/24 所在单元的位置,访问对应的一级索引表单元.一级索引表单元内容为(1,0,50),单元第 1 个 bit 为 1,所以单元后两个域分别为 BN(0),BO(50).把 BN 和 BO 交给 FPGA 模块.FPGA 模块通过分组的源地址、目的地址和 BN,得到选择的下一跳索引 index(1),并最终计算出要访问二级索引表单元位置(BA+BO+1)=51,访问二级索引表第 51 单元得到映射表的索引(26).访问映射表的第 26 单元,得到转发信息(3.1.1.1,4).

2.3 相关问题

本方案查找过程可分为 4 个阶段:TCAM 芯片查找、一级索引表访问、二级索引表访问、映射表访问.4 个阶段互相独立,不存在共享资源冲突,可以使用流水线技术提高查找速度.比如,TCAM 芯片采用 100M 次/秒的芯片、其他部分的存储都采用 10ns 的 SRAM,那么使用流水线技术,可以获得 100M 次/s 的查找速度.

方案对 IPv6 具有很好的扩展性.由于一级索引表与二级索引表都采用索引方式对路由进行存储,索引与协议是无关的,所以方案从 IPv4 升级到 IPv6,只需要扩充映射表单元数据宽度就可以满足要求.

方案中下一跳信息的存储所需空间是比较小的.对多下一跳的存储主要采用索引的形式,这极大地减小了存储空间.

3 路由更新

在 Internet 网络中,拓扑是动态变化的,路由也是动态改变的.路由器为了保证分组的转发正确,必须快速地对路由改变作出反映,及时地把新路由加入到路由表,并把过期路由从路由表中删除.路由更新的处理速度成为路由查找方案的一个重要评价指标.

为了保证方案在获得高速路由查找的同时也具备快速的路由更新,我们对方案的路由更新作了进一步的研究.路由更新过程可分为两个部分:TCAM 芯片内路由前缀的更新以及下一跳存储部分的更新.下面是路由更

新过程的伪码.

```
Update()
{
    根据路由前缀,更新 TCAM 芯片及一级索引表;
    if (路由表项具有多个下一跳)
        更新二级索引表;
    更新映射表;
}
```

3.1 TCAM芯片路由前缀的更新

3.1.1 N 子空间更新算法

为了应用 TCAM 进行最长前缀匹配查找,TCAM 芯片内的路由前缀必须按照一定的规则进行存储,这种规则我们称为约束.前缀长度约束和前缀链约束是两类不同的约束[9].前缀长度约束是指前缀长度长的前缀在 TCAM 内的存储地址一定要比前缀长度短的前缀低.前缀链约束基于 Trie 树结构,Trie 树中从根结点到叶子结点的路径称为链.前缀链约束是指链上越靠近叶子的前缀应该存储在越低的地址中.

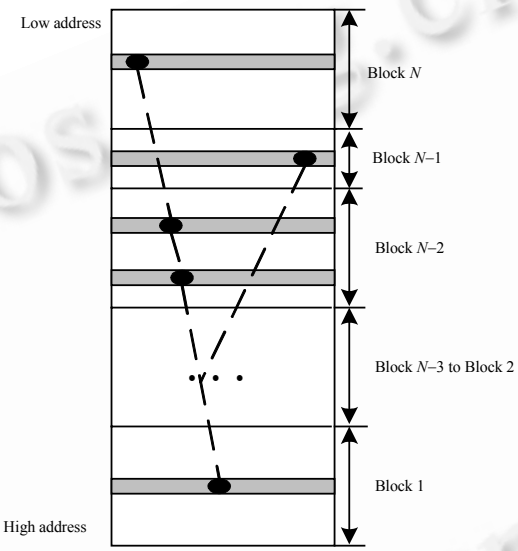

Fig.3 Subspace partition in TCAM
图 3 TCAM 内子空间划分

一般来说,基于前缀链约束在 TCAM 内进行路由前缀更新具有更好的性能.我们提出的 N 子空间更新算法就基于前缀链约束.把 TCAM 内的空间划分为 N 个子空间,分别为 $\{S_1,S_2,S_3,\ldots,S_{N-1},S_N\}$;每个子空间关联一个前缀区间,分别为 $\{L_1=[0,l_1], L_2=[l_1+1,l_2],\ldots,L_{N-1}=[l_{N-2}+1,l_{N-1}],L_N=[l_{N-1}+1,W]\}$,其中 $l_1,l_2,\ldots,l_{N-1}$ 为 $[0,W]$ 之间的整数,并且满足 $l_1<l_2<\ldots<l_{N-1}$;这些区间把路由前缀长度区间 $[0,W]$ 分成了连续的 N 段(W 为 IP 地址长度,也是最大路由前缀长度).如果某个前缀的前缀长度属于前缀区间 L_k,则该前缀在 TCAM 内应存储在对应的 S_k 子空间中.图 3 为 TCAM 内子空间划分的示意图.图中灰色条框表示 TCAM 内的一个地址单元.虚线表示前缀在 Tries 树中的链关系.

子空间内空闲的空间位于低地址端,与把空闲空间放在高地址端相比,可以有效地减少更新时移动的次数.链中越靠近叶子的结点,结点的密度越大,如果空闲空间放在高地址端,当添加靠近叶子结点时,它们的父结点需要频繁地进行移动操作,这极大地增加了更新时移动操作的次数.当进行前缀更新时,首先根据前缀的长度判断前缀所属的子空间,然后在对应的子空间内对该前缀进行添加和删除操作.下面我们对子空间内前缀的添加和删除作进一步说明.

- 子空间内前缀添加

图 4 为子空间内前缀添加示意图.为了基于链完成前缀更新,我们需要在 Trie 树中为每个前缀结点 p 维护一个指针 hcld(p),hcld(p)指向 p 结点多个前缀子结点中,在 TCAM 中存储位置最低的结点.如图 4 所示,新加入的前缀结点 p 要加入 TCAM 中,具体步骤如下:

(1) 设当前结点 $q=p$;

(2) 如果 hcld(q)结点前缀位于子空间内,则跳至(3);否则,跳至(4);

(3) 把 q 结点移至 hcld(q)结点位置,置 q=hcld(q),转至(2);

(4) 把 hcld(q)移至空闲区间.

• 子空间内前缀删除

图 5 为子空间内前缀删除的示意图.为了基于链进行前缀的删除,我们需要为 Trie 树中每个结点 p 维护一个 parent(p)指针,它指向 p 结点 Trie 树中的前缀父结点,同时我们还需要记录子空间中最靠近空闲空间的前缀结点.如图 5 所示,结点 p 为要删除的结点,q_0 为最靠近空闲空间的结点,我们按照下面的步骤删除结点 p:

(1) 设当前结点 $q=q_0$;

(2) 如果 parent(q)结点位置高于删除结点 p 的位置,则跳至(4);否则,跳至(3);

(3) 移动 q 结点至 parent(q)结点位置,置 q=parent(q),转至(2);

(4) 将 q 结点前缀写入 p 结点位置.

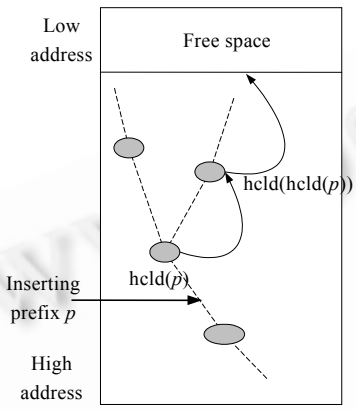

Fig.4 Inserting prefix

图 4 增加前缀

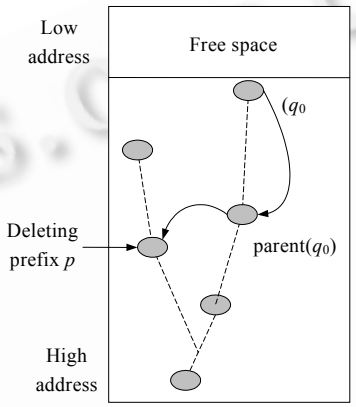

Fig.5 Deleting prefix

图 5 删除前缀

假设 Trie 树中最大链长为 D,最好情况下,更新算法的复杂度为(D/N);最差情况下,更新复杂度为 D.如果我们把子空间的空闲区间放在中间,更新算法的性能会进一步提高,最好情况下为 D/2N,最差情况下为 D/2.

图 4 和图 5 中空闲空间放在低地址端,当子空间的空闲空间被耗尽时,该子空间无法再加入新的前缀.此时,我们采用两种策略处理.第 1,如果该子空间不是 Block N(即位于最低地址端的子空间),则和紧连的低地址端子空间进行合并.比如,Block N–1 被充满,则 Block N–1 需要和 Block N 进行合并,形成一个新的子空间.合并后,子空间个数从 N 个变为 N–1 个.第 2,如果该子空间是 Block N(即位于最低地址端的子空间),则 Block N 和 Block N–1 共用 Block N–1 的空闲空间.此时,空闲空间位于 Block N 的高地址端,对应子空间增加前缀和删除前缀的方法与图 4 和图 5 类似,这里就不再赘述.

从复杂度的分析可以看出,N 越大,N 子空间算法性能越好,当 N=W 时(即子空间个数等于最大前缀长度),算法的更新复杂度达到最小.但对于实际路由表[10],N=3 时再继续增大 N,算法平均性能改善很小,这一点我们会在实验部分加以说明,而且增加 N 还会导致子空间存储空间分配达不到最优、子空间频繁发生合并等问题.下面,我们引入一个 N 子空间算法性能的评价函数以及一种确定 N 子空间最佳前缀区间划分的方法,利用它们,我们可以确定最佳前缀区间的划分,同时也可以选择合适的 N 值.

3.1.2 N 子空间的前缀区间划分

N 子空间更新算法复杂度由 D 和 N 决定,但算法的平均性能与 N 个子空间前缀区间的划分有直接关系,不合适的划分将导致 TCAM 内平均更新性能的下降.为了对这种影响进行量化,我们用子空间划分评价函数 T(W,N)对划分进行评价.T(W,N)表示给定一个路由表,在 N 个子空间划分下,更新前缀长度在[0,W]之间的路由,TCAM 芯片内移动操作和写入操作的总数目.当 T(W,N)达到最小时,我们认为此时 N 个子空间的划分是最优的.

假设 N 个子空间中第 N 个子空间的前缀区间为 L_N=(i,W),对[0,W]区间内的前缀更新,分为两个部分:[0,i–1]

区间和[i,W]区间.[i,W]区间前缀在子空间 L_N 内更新,[0,i–1]区间的前缀在 L_1 到 L_{N-1} 这 N–1 个子空间内更新.由此我们可以得到 $T(W,N)$的递归表达式:

$$T(W,N)=T(i-1,N-1)+F(i,W) \tag{1}$$

当 $T(W,N)$最小时,我们会进一步得到式(2):

$$T_{\min}(W,N)=\underset{i=N\ \text{to}\ W}{\text{Min}}(T_{\min}(i-1,N-1)+F(i,W)) \tag{2}$$

在式(1)和式(2)中,$F(i,W)$表示在区间[i,W]内的前缀在 L_N 子空间内更新所需操作的数目.$T(i,N-1)$表示在区间[0,i–1]内的前缀在 L_1 到 L_{N-1} 这 N–1 个子空间内更新时所需的操作数目.

对于式(2),我们可以推广得到更一般的表达式(3):

$$T_{\min}(y,x)=\underset{i=x\ \text{to}\ y}{\text{Min}}(T_{\min}(i-1,x-1)+F(i,y)) \tag{3}$$

在式(3)中,y 属于区间[1,W],x 属于区间[1,N].$T_{\min}(y,x)$表示对给定的路由表,在 x 个子空间划分情况下,更新前缀长度在[0,y]之间的路由,所需的最小移动和写入操作数.$T_{\min}(i-1,x-1)$表示在区间[0,i–1]内的前缀,在 L_1 到 L_{x-1} 这(x–1)个子空间内更新时所需的操作.$F(i,y)$表示在区间[i,y]内的前缀,在 L_x 子空间内更新所需操作的数目.

式(3)是通用的表达式,它可以计算任意取值的 $T_{\min}(x,y)$.式(3)是我们计算 $T_{\min}(W,N)$的基础.我们采用下面的方法计算得到 $T_{\min}(W,N)$,从而得到最佳的前缀区间划分.首先计算初始值 $T_{\min}(k_1,1)(k_1=1\ \text{to}\ W)$,然后再利用式(3)计算 $T_{\min}(k_2,2)(k_2=2\ \text{to}\ W)$,依此类推,最后计算 $T_{\min}(W,N)$,计算过程共需要计算$((2\times W-N+2)\times(N-1)/2+1)$个 $T_{\min}$ 值.

对 $F(i,W)$值的计算除了考虑路由前缀在 Trie 树中的位置之外,还要考虑路由前缀插入的顺序.比如,Trie 树中某条链上有两个路由前缀,如果先加入靠近根节点的前缀,再加入远离根节点的前缀,则不需要移动操作,只需要写入操作,所用操作数为 2;如果先加入远离根结点的前缀,再加入靠近根结点的前缀,则需要 1 次移动操作,所用总操作数为 3.在实际计算中,$F(i,W)$可选为可能的最大操作数.

在计算得到 $T_{\min}(W,N)$的同时,我们也可以得到 N 个子空间对应的前缀区间.为了保证空间的充分利用,N 个子空间的大小可依据各自空间内部路由前缀占路由表的比例来进行分配.

利用式(3),我们还可以比较不同 N 值对应的 $T_{\min}(W,N)$,从而选择一个大小合适的 N 值.

3.1.3 下一跳存储部分的更新

一级索引表的更新是直观的.对于新加入的路由表项,需要根据路由表项下一跳的个数来填写对应存储单元的内容.存储单元中的 BN,BO,NHN 以及映射表索引域需要根据二级索引表和映射表更新的结果来填写.对于删除的路由表项,根据一级索引表单元中的 BN,BO,NHN 以及映射表索引域对二级索引表和映射表进行更新,同时释放该路由表项在一级索引表的存储单元.对于 TCAM 芯片中路由前缀的移动操作,对应一级索引表中的单元需要进行相应的移动操作.比如,TCAM 芯片中某个路由前缀从位置 k_1 移动到位置 k_2,那么一级索引表中 k_1 单元也要移动到 k_2 单元.

为了有效地利用二级索引表的空间,我们用链表结构管理空间的分配.块 1、块 2、块 3 和块 4 每个块内部空间的分配都用一个链表结构来管理.在块 1 中,空闲空间两个一组用链表连接起来.增加路由时,从链表头部取下一组空闲空间分配给新的路由,删除路由时,把该路由占用的两个连续空间组成一组,加入到链表的头部,图 6(a)为块 1 的链表结构.块 2、块 3 的空间管理与块 1 类似,不同的是,空闲空间分别以 3 个和 4 个为一组.块 4 中路由的下一跳数目不定,所以它的管理复杂一些.在块 4 中,链表把一段段连续的空闲空间按照它们的地址顺序连接起来,链表中的表项记录块 4 内一段连续空间

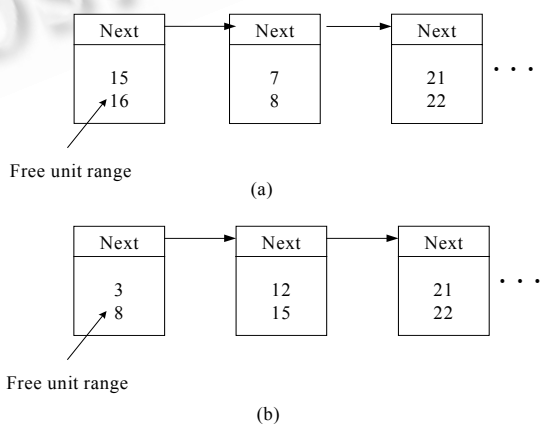

Fig.6 Block management list

图 6 块管理链表

的首尾地址.初始状态,块 4 的链表中只有一个表项,它代表整个块空间.增加路由时,依次搜索链表中的表项,找到第 1 个可以满足该路由下一跳存储所需空间的表项返回,在这段空间内,占用该路由要求的空间,把剩余空间重新加入到链表中.删除路由表项时,直接把路由占用的空间加入到链表,如果可以和链表中前后表项的空间合并,则需要合并成一个表项,然后把这个表项重新加入链表中,图 6(b)为块 4 的链表结构.

映射表的更新需要为每个映射表单元维护一个引用计数.增加路由表项时,如果该路由表项的(下一跳 IP,出端口)信息在映射表中不存在,则找到映射表的第 1 个空闲单元,加入新的(下一跳 IP,出端口)信息,同时该单元的引用计数加 1;如果对应路由表项在映射表中存在,则把单元的引用计数直接加 1.删除路由时,把相应的映射表单元引用计数减 1,如果该单元引用计数为 0,则置此单元为空闲单元.

4 路由压缩

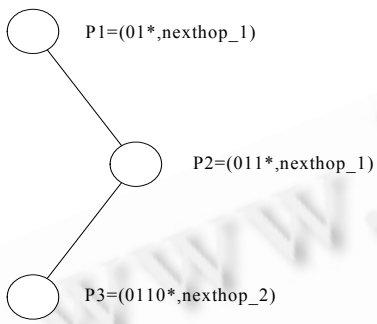

Fig.7 Routing compaction
图 7 路由压缩

为了减少 TCAM 芯片的成本和功耗,我们对 TCAM 路由表进行压缩.压缩的基本思想是向 TCAM 内更新前缀时,判断此前缀是否冗余,如果冗余则不更新至 TCAM 中,这样就减少了 TCAM 空间的使用.有效而快速地判断路由前缀是否冗余是路由压缩的关键.

我们基于 Trie 树结构作冗余前缀的判断.如图 7 所示,Trie 树中,前缀 P1 为路由前缀 P2 的父结点,同时 P1 具有和 P2 相同的下一跳信息,对于分组转发来说,路由查找结果为 P1 或为 P2,最终的转发结果是一样的.前缀 P2 存在与否并不影响分组转发的结果,因此前缀 P2 是冗余的,它不必更新至 TCAM 中.在路由表中删除所有的类似 P2 的冗余前缀,就实现了对 TCAM 路由表的有效压缩.

路由压缩由于也是基于 Trie 树结构的,它可以与前面提到的 N 子空间更新算法同时使用,在路由前缀更新时,首先判断此前缀是否冗余,如果冗余则不更新至 TCAM 中,由于冗余判断的操作很简单,它不会影响 TCAM 的更新速度.

5 实 验

我们对 N 子空间更新算法与 CAO_OPT 和 PLO_OPT 算法[9]的更新性能进行比较.比较的内容为更新前缀所需的平均移动次数(简单起见,写入操作也算作一次移动操作).对于 N 子空间更新算法,分别选取 N=3 和 N=4.进行 N 子空间划分时,选用 Mae-East 路由表作为划分依据的路由表.利用第 3.1.2 节介绍的方法,可以求得 N=3 时,对应 3 个子空间的前缀区间分别为 L_1=(0,18),L_2=(19,23),L_3=(24,32);N=4 时,对应的 4 个子空间的前缀区间分别为 L_1=(0,18),L_2=(19,23),L_3=(24),L_4=(25,34).

我们选择 5 个路由表进行比较.Mae-East 和 Mae-West 是实际 Internet 网络中的路由表,它们的规模较小.随机路由表 1、随机路由表 2 和随机路由表 3 是根据 Mae-East 和 Mae-West 路由表中前缀长度的分布情况随机生成的,它们的规模较大.选择 3 个规模较大的路由表主要是考查,当路由表规模进一步增加时,也就是在未来大规模的 Internet 路由表的条件下,各个算法的更新性能.

Table 2 Update performance comparison for different algorithms

表 2 不同算法更新性能比较

Routing table	Mae-East	Mae-West	Random 1	Random 2	Random 3
Routing number	22K	34K	500K	1M	2M
N subspace algorithm (N=3)	1.01	1.02	1.12	1.18	1.26
N subspace algorithm (N=4)	1.01	1.02	1.09	1.12	1.16
CAO_OPT algorithm	1.24	1.30	1.58	1.96	2.46
PLO_OPT algorithm	3.35	3.41	3.37	3.34	3.33

表 2 是对各个算法平均更新性能的比较结果.可以看出,PLO_OPT 算法基于前缀长度约束,更新性能最差,它只与路由表前缀长度分布的情况有关,与路由表规模无关,路由表规模增大时,PLO_OPT 的更新性能不受影响.CAO_OPT 和 N 子空间更新算法基于前缀链约束,它们的性能明显优于 PLO_OPT 算法.当路由表规模增大时,Trie 树中结点数目增多,链长度增加,相应地,CAO_OPT 和 N 子空间的更新性能也随之下降.当路由表规模比较小时,比如 Mae_East 和 Mae_West 路由表,CAO_OPT 和 N 子空间更新算法性能比较接近,都可以达到近似 $O(1)$的更新复杂度.当路由表规模增大时,CAO_OPT 性能下降比较快,对于 2M 规模的随机路由表 3,CAO_OPT 平均更新移动次数为 2.46,对比 Mae-East 路由表,性能下降了 150%左右;N 子空间算法性能也在下降,但下降速度较慢,对于随机路由表 3,N 子空间算法(N=4)平均更新移动次数为 1.16,对比 Mae-East 路由表,性能下降了 15%.可见,当路由表规模比较大时,N 子空间算法比 CAO_OPT 算法具有更好的更新性能.

N=4 时,N 子空间算法性能优于 N=3 时的算法性能,但二者相差不多,这主要是因为前缀长度大于 24 的路由不多,这使得 N=3 时 L_3 子空间内的前缀与 N=4 时 L_3 和 L_4 子空间的前缀总的更新移动次数相当,所以算法的平均更新性能也相差不多.

Table 3 Routing compaction

表 3 路由压缩

Routing table	Mae-East	Mae-West	Aads	Paix	Pacific Bell
Route number (K)	22	34	30	16	42
Reduced route (%)	19.0	19.5	21.1	22.2	24.5
Waste time (ms)	51	81	70	41	100

表 3 是应用 5 部分的压缩方法对 5 个实际路由表进行压缩的结果.时间测试使用的平台为 PC 机,它的 CPU 为 PIII933,内存大小为 256M.可以看出,我们使用的压缩方法是有效的,对于实际的路由表可以达到 20%左右的压缩比,同时,压缩所需的时间也是很短的,每毫秒大约可以完成 2.5K 条路由的压缩.因此,我们可以把压缩算法和 N 子空间更新算法结合起来使用,这样我们既可以实现 TCAM 的高速更新,又可以对路由表进行有效的压缩,从而使 TCAM 技术的 3 个缺点得到不同程度的克服.

Table 4 Storage space for next hops

表 4 下一跳存储空间

Routing table	Mae-East	Mae-West	Aads	Paix	Pacific Bell
Route number (K)	22	34	30	16	42
Nexthop storage space (K)	63	110	74	50	110

表 4 是用图 1 方案实现多下一跳路由查找时,各个路由表下一跳部分存储空间的比较,具体来说,就是图 1 中一级索引表(16 bits 宽)、二级索引表(8 bits 宽)和映射表(64 bits 宽)所占空间的比较.可以看出,下一跳部分的存储空间大概是路由表规模的 3 倍左右.其中一级索引表所占空间最大,为路由表规模的 2 倍,如果想进一步减少一级索引表空间,可以根据实际情况缩短单元数据宽度;二级索引表所占空间约为路由表规模的 1 倍,二级索引表的空间取决于一级索引表的深度以及多下一跳路由的比例;映射表所占空间为 256×64 bits,这个空间和一级索引表和二级索引表相比可以忽略.

6 结 论

本文基于 TCAM 技术,提出了一个完整的支持多下一跳查找的高速路由查找方案.方案使用两级索引表结构,实现了对多下一跳路由的存储.该方案使用映射表存储实际的下一跳转发信息,有效地减少了存储空间.为了解决 TCAM 技术的 3 个缺点,方案中提出了一个对实际路由表可以达到近似 $O(1)$更新复杂度的 TCAM 更新算法,同时还应用一个有效的路由压缩技术对 TCAM 路由表进行了压缩.

致谢 审稿人对本文提出了很多有益的建议,在此表示感谢.

References:

[1] Fuller V, Li T, Yu J, Varadhan K. Classless inter-domain routing (CIDR): An address assignment and aggregation strategy. RFC1519, 1993.

[2] Hinden R, Deering S. IP version 6 addressing architecture. RFC 2373, 1998.

[3] DegerMark M, Brodnik A, Carlsson S, Pink S. Small forwarding tables for fast routing lookups. In: Handley M, *et al*. ed. Proc. of the ACM SIGCOMM. New York: ACM Press, 1997. 3~14.

[4] Gupta P, Lin S, Mckeown N. Routing lookups in hardware at memory access speeds. In: Charny A, ed. Proc. of the IEEE INFOCOM. San Franciso: IEEE Communications Society, 1998.

[5] Waldvogel M, Varghese G, Turner J, Plattner B. Scalable high speed IP routing lookups. In: Handley M, *et al*. ed. Proc. of the ACM SIGCOMM. New York: ACM Press, 1997. 25~36.

[6] Lampson B, Srinivasan V, Varghese G. IP lookups using multiway and multicolumn search. IEEE/ACM Trans. on Networking, 1999,7(3):324~334.

[7] Nilsson S, Karlsson G. IP address lookup using LC-Tries. IEEE Journal on Selected Areas in Communications, 1999, 17(6):1083~1092.

[8] Liu H. Routing table compaction in ternary CAM. IEEE Micro, 2002,22(1):58~64.

[9] Shah D, Gupta P. Fast updating algorithms for TCAMs. IEEE Micro, 2001,21(1):36~47.

[10] Michigan University. Merit network. Internet performance and analysis project. http://www.merit.edu/~ipma

第 2 届全国智能科学与技术教育学术研讨会(第 1 轮)

征 文 通 知

为积极引导和大力推进我国智能科学与技术教育的健康、快速发展，中国人工智能学会教育工作委员会决定于 2004 年 9 月 24~26 日（暂定）在北京召开第 2 届全国智能科学与技术教育学术研讨会。会议由中国人工智能学会教育工作委员会主办，首都师范大学承办，协办单位包括北京大学、北京科技大学、北京工商大学。大会将从录用的论文中，挑选一部分优秀论文推荐到《计算机工程与应用》杂志正刊上发表。其余论文将在《计算机工程与应用》增刊上发表。

一、 会议征文范围

1. 我国智能科学与技术教育的机遇、困难及对策；
2. “智能科学与技术”学科的学科内涵及研究对象；
3. “智能科学与技术”学科的二级学科设置及培养要求；
4. “智能科学与技术”本科专业的培养方案；
5. “智能科学与技术”本科专业的课程体系和教学要求；
6. “智能科学与技术”本科专业的实验教学体系和实践环节；
7. “智能科学与技术”学科（专业）的教材体系及建设；
8. 人工智能课程的教学要求及教材建设；
9. 智能教学软件和智能教学系统；
10. 与智能科学技术有关的基础、理论、技术研究和各种应用研究；
11. 新课程标准执行后，在中学开展人工智能教育的研究与探讨；
12. 推进我国智能科学与技术教育事业发展的协调和组织工作。

二、 大会征文要求

论文必须未公开发表过，字数一般不超过 6000 字（含图表）；论文应符合科技论文的撰写要求，并必须有英文的题目、摘要和关键词；每篇论文都要分别提供激光打印文本和电子文档各一份，并一律采用 A4 纸型、1.5 倍行距。论文第一作者必须提供详细通信地址、邮编、电话和 E-mail 地址。

三、 论文方面的重要日期

论文截稿日期：2004 年 6 月 30 日（以邮戳为准） 发出录用通知的日期：2004 年 7 月 31 日

四、 投稿及会议联系地址

单位：首都师范大学信息工程学院（北京市西三环北路 105 号），邮编：100037

邮箱：xieda89@263.net geqp@mail.cnu.edu.cn wangwansen@263.net

联系人：010-68901048（谢达） 010-68901041（葛庆平） 010-68903443（王万森）