2004 年全国博士生学术论坛
四川成都 电子科技大学

一种新的快速地址查找算法[1]

彭艳兵，龚俭

东南大学计算机科学与工程系，江苏南京，210096，ybpeng@njnet.edu.cn

摘　要：　快速地址查找是路由器最主要的工作之一，路由器根据地址查询的结果决定报文的转发方向，因而快速地址查询算法对路由器性能的影响最重要。现有的快速地址查询算法的空间复杂度和时间复杂度已有很大的改进，如已经广泛使用的最长地址匹配，已经把查找次数降低到了最多 32 次。如果想超越这个方法，需要从新的角度来寻找突破。本文从地址空间统计和分布的角度出发，发现在一定的活跃地址分布下，报文的平均查找次数有可能比最长地址匹配方法小。因此本文提出一种新的 IP 地址查询算法，在较低的时间复杂度和较低的空间复杂度的要求下，能够具有很好的性能，算法非常简单，易于硬件实现。

关键词：　快速地址查询；最长前缀匹配；统计优化

A New Type of Fast Address Searching Algorithm

Yanbing Peng, Jian Gong

Southeast University, Computer Science and Engineering Department, Nanjing, Jiangsu，210096, ybpeng@njnet.edu.cn

Abstract:　It is an important work for the router that fast address searching in the limited time. The router decides the forwarding direction by the result of those searches. The current address searching algorithms have been great improved in time complexity and spatial complexity, e.g., as the widest deployed algorithm, the longest prefix match of address, decreased the search counts no more than 32 times. A new visual angle shall be introduced into the fast search algorithm if an exceeding is done towards this method. It was found that under some active IP address distribution of traffic load, the linear router table search can be optimized, and its performance could be compared with the longest prefix matching algorithm. With this method, a new type of IP address searching algorithm was proposed to provide better performance than the longest prefix matching algorithm.

key words:　Fast address searching; the longest prefix matching; Optimizing by Statistics.

1　引言

对于高速主干路由器，其路由表的选项一般在 150k 左右，最高的可达到 500k 左右。主干网络的带宽一般在 2.5Gbps 以上，10Gbps 以上的线路已经在使用。因此，每秒钟路由器需要处理的报文在 1M/s 以上，每报文的处理时间要求小于 1μs。高性能的路由器要求每秒处理 4M/s 以上的报文，链路的带宽超

[1] 基金资助：本文受国家 973 计划 2003CB314803、NNSFC No. 90104031 和 863 计划 No. 2001AA112060.支持
作者简介：彭艳兵（1975-），男，湖北洪湖人，东南大学计算机系在读博士

过 2Gbps。高速的主干网络流量对路由表的查询提出了很高的要求，因此快速地址算法应运而生。

针对高速路由查找算法需要考虑的问题，文献[1]提出有如下问题需要解决：新颖的算法要比现有的算法有更高的性能，或者更小的内存占用；高速存储访问是提高性能的一个关键因素；容易用硬件实现。文献[1] 提出了几个快速地址查询算法，并与其它算法进行了比较。

现有的路由查询算法主要有基于树的查询、基于比较的线性查询、基于哈希的查询、结合哈希和树的查询。基于哈希的查询在性能上不完善，有些如线性比较查询在时间和空间上消耗太多，无法利用快速 SRAM 等硬件带来的好处[1]。陆晟，龚俭[2] 提出了一种表达能力很强，但是性能平均的报文分类算法——不相交树算法，该算法的时间复杂度和空间复杂度并不是最优的，但由于其能够表达复杂的分类而应用于需要强表达能力的系统如 IDS 上。TCAM[2]硬件算法的速度最高，但是其高昂的成本使得它只能应用于对速度非常挑剔的环境里。因此本文主要针对最长前缀匹配树算法来展开讨论。

文献[1]提出了一种基于最长前缀匹配的快速路由查找算法，查找快速且空间占用低；文献[3] 给出了最长前缀匹配基于 Bloom Filter 的改进，但是性能上的改进不大。最长前缀匹配地址查找算法的时间和空间复杂度如下：对于任意的 IP 地址，最多比较 32 次，文献[1]中认为可以忽略，但本文主要针对这个数字展开我们的工作；对于 N 个长度基本为 H 的前缀，添加前缀的计算的复杂度为 O(H)，因此计算复杂度为 O(N×H)。对于压缩的前缀表的查询操作为 O(1)，因此总体复杂度为 O(N×H)；内存耗费为 M=3×N×H，空间复杂度为 O(N×H)。

2　改进

但是由于最长前缀匹配算法和很多其它已知的算法一样都没有考虑到实际网络中活跃地址的分布，所以受到局限，无法对其进行进一步的优化。本文对大量的主干网流量分析后发现，活跃 IP 地址的分布服从一定的规律，如图 1 所示，图中的组数是指 IP 地址数目，其中后面一组包含的 IP 地址的数目是前面一项 IP 地址的 2 倍。该表折合到路由表项会更加集中在前面的一些网段。由于主干网流量 80%的 IP 集中分布在 15 个 IP 地址，99.9%的 IP 集中在前 5,000 个 IP[4] ，因此可以按路由表项使用频数对 IP 路由表进行排序，能够显著减少路由表访问次数，优化路由器的路由查找效率。

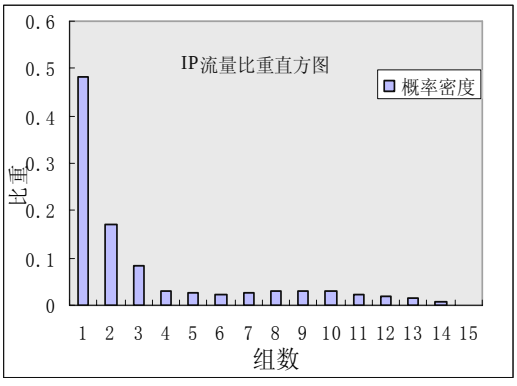

图 1 IP 分组数与整体比例*

Fig. 1 Packets counts against total rate

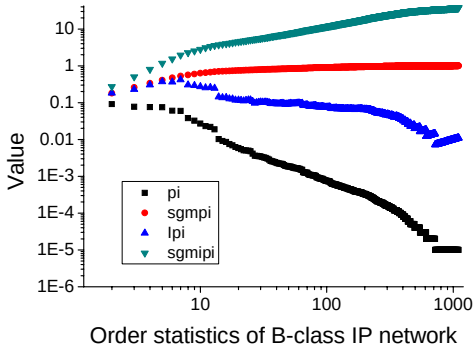

图 2 B 类网段的 IP 报文数次序统计曲线*

Fig. 2 Order Statistical Curves of IP packets Counts Class B network

pi: p_i　　sgmpi: Σp_i　　Ipi: $i*p_i$　　sgmipi: $\Sigma i*p_i$

$\max(\Sigma p_i) = 0.994$,　$\max(\Sigma i*p_i) = 37.94$,　L=1102

*：数据收集于中国教育科研网江苏省边界，图 1 为一天的数据分析，图 2 为 1 小时的数据

根据上述结论可以提出一个新的路由查询算法。针对这个规律我们提出一个测度——路由表平均查找次数，用来描述路由算法的效率：

路由表的平均查找次数 I：查询 IP 转发方向是路由表访问次数的平均值。

设某 IP 或者 IP 网段的访问概率为 p(i)，i 为其在线性路由表的位置号，也就是该路由的查找次数。

路由表长度为 n，i 项访问次数为 N_i，总访问次数 N，p(i)为 i 被访问的概率，则平均查找次数 I：

$$I = E(i) = \int_n i * p(i)di = (\sum_{i=1}^{n} N_i * i) / N = \sum_{i=1}^{n} p(i) * i$$

2.1　没进行统计优化的路由表平均查找次数 I

设访问次数为 N，在路由表 n 内找到路由的概率是 1，没有优化的路由在(0, n)内均匀分布，则第 i 项被访问的概率为 1/n；

$$I = \frac{1}{N}\sum_{i=1}^{n} N * p(i) * i = \sum_{i=1}^{n} \frac{i}{n} = \frac{(n+1)}{2}$$

当有 M 次 IP 访问时，路由器需要查找 M(n+1)/2 次路由表，计算量巨大

对于最长前缀匹配，没有线性路由表可以供我们使用，但是根据最长前缀匹配的机理，我们可以推出，对于每个到达的报文，最长前缀匹配的最大路由表访问次数是 32，最小路由表访问次数是 8。假设最长前缀为均匀分布，可知其平均访问次数为 20，则平均每个报文最长前缀匹配需要访问 20 次路由表。

2.2　统计优化的路由表

一般的路由器使用最长前缀树匹配，最多只需 32 次查找，如果我们的算法能够使得平均查找次数小于最长前缀匹配的平均查找次数，就可能有比最长前缀匹配树更好的性能。对于一个线性路由表，我们在查找比例为 d 处优化截止，若路由表项命中的次序分布为 f(i)，此时对应的查找表为 L；剩下(1-d)的查找按照普通的路由查找方法，其统计分布是 p(i)：

$$I = \frac{1}{N}(\sum_{i=1}^{L} N * f(i) * i * d + \sum_{i=L}^{n} N * p(i) * (1-d) * i) = \sum_{i=1}^{L} i * f(i) * d + \sum_{i=L}^{n} i * p(i) * (1-d)$$

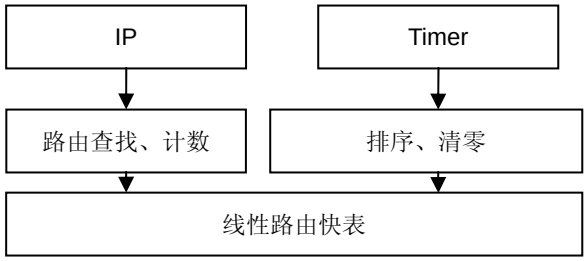

图 3　基于统计优化的路由查找框架结构

Fig. 3 IP Address searching structure based on statistical optimizing algorithm

表 1　统计路由优化算法的路由快表

Table 1 Fast Routing table of statistical optimizing IP searching algorithm

i	item	direction	counter
1	Routing item 1	Port1	count1
2	Routing item 2	Port2	count2
3	Routing item 3	Port3	Count3
…	……	……	……

根据图 2 的活跃 IP 次序分布计算的前 20 个 IP 地址的平均查找次数小于 4.61；对于 98.6%的报文，

本算法平均只要寻找 32 次就可以找到该地址，此时线性路由表的位置为 553。

基于路由表优化的算法如下

- 1）查询路由

逐项查找路由表 until(longest prefix matched){
该计算器加一，返回路由方向 }

- 2）路由表优化和计数器刷新

每 x 秒更新优先顺序一次 ;每 y 秒计数器清零一次，以避免计数器溢出时打乱优化好的计数器；[要求时间复杂度为 O(1)的算法]；

排序优化有很快捷的方法，如赛跑触线法，利用计数器的翻转作为优先顺序的判别，时间复杂度和空间复杂度都能够满足本文的要求，具体算法不在这里给出。另外时间变化模型研究显示，线性路由表里活跃的表项的顺序在很长的时间范围内变化很小，因此排序优化只需在较长的时间里做一次就可以了，其计算复杂度可以忽略。路由表的更新也就是一次路由表的查找过程，与其他 IP 地址查找的时空复杂度相同，但没有生成树的操作开销。表 1 为经过统计优化的路由表，图 3 为基于统计优化的路由查找优化算法的框架图。本算法时间复杂度为 O(I)，I 为平均路由查找次数，基本为一常数；空间复杂度为 O(L×H)，H 为匹配地址加计数器宽度，H 比其他算法长一些，这里 L<<N；代价，这一部分的算法只处理比例为 d 的报文。剩下的 1-d 部分的报文采用其他路由查找方式实现。由于 L 很小，本算法与 TCAM 硬件算法联合将会更快。

3　结论与展望：

基于前缀树的快速路由查找算法没有考虑活跃 IP 和 IP 网段的分布，因而不是统计最优的。活跃 IP 和活跃路由表的分布给我们提供了一种优化路由查询的方法，利用这种方法可以显著地降低查询计算量，并由此提出了一个新的测度——平均查询次数 I，用于评价快速路由算法。经过优化的线性路由表比没有优化的线性路由表的优点在于其极大地减少了路由表的查询次数，相应地提高了性能，使得它能够和最长前缀匹配方法相提并论。

基于平均查询次数的优化路由查询和优化算法极其简单，非常容易硬件化。缺点在于优化的路由表只能采用线性方式存放，不能采用其他压缩方式如二叉树进行；由于需要累加和排序，整体开销与最长前缀匹配相当，但活跃 IP 的长时间稳定分布能够显著减少排序的计算量，并采用一种复杂度为 1 的快速排序方法后，则本算法在性能上全面超越最长前缀匹配。

如果把路由表的所有路由项都作为查找表的开销会比较大，一种明智的方法是与其他查找算法结合，把路由命中率很小的不活跃的表项使用其他方式进行查找是一个非常好的思路，比较有前途。

参考文献

[1] Henry Hong-Yi Tzeng, On Fast Address-Lookup Algorithms, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 17, NO. 6, JUNE 1999

[2] 陆晟，龚俭，一种新的高维报文分类算法——无相交树算法，计算机学报，Vol. 26 No. 11，Nov. 2003，1502-1509

[3] Sarang Dharmapurikar Praveen Krishnamurthy David E. Taylor, Longest Prefix Matching using Bloom Filters. SIGCOMM conference 2003

[4] 程光，龚俭，丁伟，大规模互联网活动 IP 流分布研究，计算机科学，2003 年 4 月