第 35 卷 第 2 期 计 算 机 工 程 2009 年 1 月
Vol.35 No.2 Computer Engineering January 2009
·博士论文· 文章编号：1000—3428(2009)02—0004—02 文献标识码：A 中图分类号：TP311

BGP 路由查表算法的分析与改进

马海龙[1]，郭云飞[1]，陈乐然[2]
(1. 国家数字交换系统工程技术研究中心，郑州 450002；2. 解放军信息工程大学理学院，郑州 450001)

摘 要：Default-Free Zone 内的路由器拥有巨大的路由表项，为了方便实现，大多数开发者在实现过程中均采用路径压缩树的数据结构对边界网关协议路由进行存储。计算机仿真分析发现，采用路径压缩树算法会引起路由搜索时间抖动严重、存储空间利用率低，路径压缩树性能受 BGP 路由前缀的分布特性影响。引入 AVL 算法解决存在的问题，可以较好地提高路由查表的性能。
关键词：边界网关协议；路径压缩树；AVL 算法

Analysis and Improvement of BGP Route Lookup Algorithm

MA Hai-long[1], GUO Yun-fei[1], CHEN Le-ran[2]
(1. National Digital Switching System Engineering & Technological R & D Center, Zhengzhou 450002;
2. Science Institute, PLA Information Engineering University, Zhengzhou 450001)

【Abstract】There are vastly routes in Border Gateway Protocol(BGP) router located in the Default-Free Zone. Many developers adopt Path Compressed Tree(PCT) to store the BGP routes. It is found that the performance of PCT algorithm is not so good. The lookup time fluctuates and the store memory is wasted. The performances of PCT are mostly influenced by the character of data source. In order to solve those problems, AVL algorithm is introduced. Comparison results of these two algorithms manifest that the AVL algorithm can improve the lookup performance.
【Key words】Border Gateway Protocol(BGP); Path Compressed Tree(PCT); AVL algorithm

1 概述

边界网关协议(Border Gateway Protocol, BGP)[1]是用于自治系统之间进行路由信息交互的域间路由协议。Default-Free Zone内路由器应当拥有到达所有IP地址的路由，因此，其路由表项信息量很大。据最新统计，Internet核心路由器上至少拥有 250 000 条路由[2]。考虑到BGP拥有如此多的路由信息，路由查表算法应当首先考虑更新复杂度，兼顾存储复杂度和查找速度，同时要支持无类域间路由(Classless Inter-Domain Routing, CIDR)和IPv6。为了实现的简易，较多的BGP开发者(如Zebra)采用路径压缩树(Path-Compressed Trie, PCT)的数据结构。

路由查表算法对 BGP 的影响主要表现在 2 个方面：BGP 的稳定性和 BGP 的可扩展性。网络的收敛速度取决于路由更新时间，而路由的更新时间很大程度上取决于路由查表的速度；BGP 路由表占用路由器的内存大小则决定了 BGP 的可扩展性。如果 BGP 的路由表项占去了路由器的大部分内存，这将严重影响路由器的性能。因此，本文对 PCT 的路由查表速度及占用的内存进行了性能分析。

2 路由查表算法的分析

2.1 路由查表时间分析

采用 2.60 GHz Pentium 4 CPU、512 MB内存的计算机完成算法的性能分析，并下载了AS33332005/02/01 上的BGP路由表信息[3]。当时，此AS内FIB中拥有的BGP表项信息共有 154 293 项。采用路径压缩树的方法来存储，生成了 287 385 个节点，叶子节点有 140 753 个，二叉树的最大搜索深度为 27，最小搜索深度为 5，平均搜索深度为 21。

从文献[4]中可知，成功和不成功搜索花费几乎同样的时间，因此，为了提高分析结果的可参考性，在仿真分析过程中，每次搜索 10 000 条路由，并且循环操作若干次，每次循环过程中搜索的路由是不同的。

本文设计了如下的方法来探寻路由查找所消耗的时间问题：从potaroo.net网站[3]上下载BGP路由表，利用计算机随机抽取了 10 000 条路由，采用二叉搜索的方法在PCT中查找这些节点。循环操作 50 次，得到了前缀节点查找算法所消耗时间的统计信息，如图 1 所示。

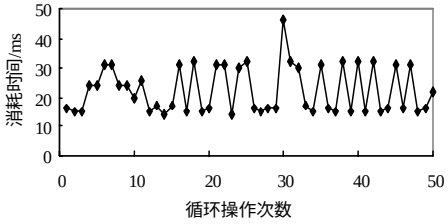

图 1 前缀查找算法消耗的时间

从图 1 可以看出，搜索 10 000 条路由所需要的时间在 32 ms 以内(舍弃一个特殊点)，但搜索时间非常不稳定，变化范围从 14 ms~32 ms，其均值为 21.776 0 ms，方差为 7.177 8。忽略计算机在执行相同代码时的时间差异，推断造成搜索时间抖动的原因是前缀节点存储深度的不同。

为了验证推断的正确性，下载了 6 个AS上的路由表[3]，下文以这些数据为对象展开分析。针对这 6 个AS中的数据，重点分析了前缀节点在路径压缩树中存储时的最大深度、最

基金项目：国家“973”计划基金资助项目(2007CB307100)
作者简介：马海龙(1980－)，男，讲师、博士研究生，主研方向：域间路由协议，下一代互联网；郭云飞，教授、博士生导师；陈乐然，讲师
收稿日期：2008-06-13 **E-mail：**longmanclear@163.com

小深度、平均深度及与理论值的差异，其中的理论值由下面的定理[4]得出：搜索任何二叉树的每个节点需要 2 次比较，并且二叉树最优高度是lb$N$。

经过相应的运算分析，得到图 2 所示的结果。

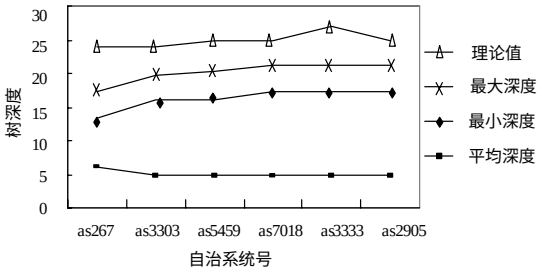

图 2　前缀节点的存储深度

在利用路径压缩树存储前缀节点时，节点的存储深度存在很大的差异，而且平均深度比最优深度长 6。正是这些差异造成了前缀搜索时间的抖动。

通过上面的分析可知，采用路径压缩树结构进行前缀节点的查找存储所消耗的时间非常不稳定，最坏情况下消耗的时间是最好情况下的 2 倍，查找时间的抖动太大。造成这种情况的主要原因有 2 个：(1)节点的深度分布非常不均匀；(2)节点的最大深度比最优深度大得多。查找时间分布的不稳定性必然会引起报文的更新速度及路由收敛时间的抖动，进一步影响整个 BGP 系统的可扩展性和稳定性。

2.2　路由表大小分析

采用路径压缩树的算法生成的路由表是一个完全二叉树，每个节点主要存储前缀、前缀长度和属性信息。由于属性信息依附在二叉树的节点上，因此本文不考虑它所占的存储空间大小。并将二叉树中存储路由表前缀的节点称为有效节点，其他节点称为无效节点(null node)。

路由表的大小取决于前缀数量以及存储这些前缀所需要的节点数。仍对上述 6 个 AS 上的路由表进行分析，利用路径压缩树数据结构保存前缀信息，得到了前缀数量、节点数及无效节点数之间的统计信息，列入表 1 中。

表 1　节点数目统计

AS 号	路由总数	节点总数	无效节点	无效节点比例/(%)
AS267 NETHER-2	9 799	18 640	8 841	47.43
AS3303 SWISSCOM	70 015	132 027	61 550	46.62
AS5459 LINX-AS	74 653	139 824	64 961	46.46
AS7018 ATTW	151 895	283 603	131 439	46.35
AS3333 RIPE-NCC-AS	156 561	291 803	135 021	46.27
AS2905 UUNET-ZA	161 158	301 454	140 067	46.46

从无效节点占节点总数的百分比来看，用路径压缩树存储地址前缀生成的无效节点数约是节点总数的一半，即在一棵拥有 2N 个节点的路径压缩树中，无效节点数约为 N 个，这样导致存储空间的利用率只有 53%左右。随着路由表中有效节点的增加，无效节点也会成倍增长，这将导致系统的可扩展性变差。在 BGP 系统的实现中，每个节点需用 60 Byte 保存前缀以及指针信息，如果存储 156 561 条路由，生成的路由表约占 17 MB 内存，其中 8 MB 是无效内容；而且 BGP 的路由表还要在路由表管理(Routing Table Management, RTM)模块中保存，造成了存储空间的进一步浪费。

3　路由查表算法的改进

路径压缩树前缀节点存储深度的不均匀导致路由搜索时间的严重抖动，而且前缀信息主要存储于叶子节点上，无效节点的数目与有效节点的数目几乎相当，使存储空间的利用率只有 53%左右，造成了存储空间的浪费。因此，采用路径压缩树数据结构会导致 BGP 系统的可扩展性和稳定性变差。

3.1　数据结构的选择

分析Oregon[2]提供的BGP路由表得到了BGP路由前缀随掩码长度的分布，如图 3 所示，从中可以看出路由前缀的分布与掩码长度有关，多数路由前缀都集中在 16~24 之间。由于PCT算法的性能对数据源有很大的依赖性，因此路由前缀分布的不均匀必然导致PCT算法查表时间不均匀、存储空间浪费等情况。因此，新的查表算法必须克服路径压缩树结构的缺点，尽量保证新算法的特性不依赖于数据源的分布特点，保证前缀节点存储深度均匀，无效节点尽可能少，更新操作简单，查找速度稳定，仍然支持CIDR和IPv6 地址，还必须是二叉树的结构。

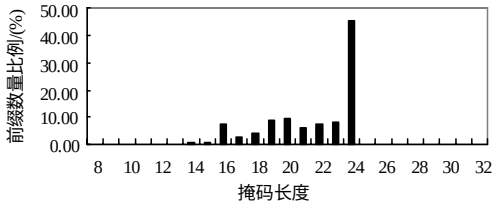

图 3　BGP 路由前缀掩码长度与前缀数量比例

平衡二叉树(AVL树)保证能在$O(\log N)$时间内完成插入和删除等操作，；新增和删除节点时能够保持树的平衡状态，它的高度不会超过1.44lb$N$ 。这样即使在最坏的情况下，AVL树行为的效率也不会低于一棵随机二叉搜索树行为的效　率[4]。经验表明，在几乎所有的情况下，AVL树搜索的时间长度接近于 lb$N$ ，这样，AVL树的行为近似于理想的、完全平衡的二叉树的行为。

AVL 树是一棵二叉搜索树，树的根节点下左右子树的高度差别至多是 1，并且左右子树也是 AVL 树。AVL 树的每个节点除了存储前缀信息外还要存储平衡因子，平衡因子用于指示当前节点的左右子树高度之间的关系，以便在进行节点的添加和删除时调整树的结构。可以看到，AVL 树数据结构能够较好地克服 PCT 树的缺点。因此，将路由表数据结构改为 AVL 树可以参照文献[5]编写相应代码。

3.2　PCT 和 AVL 的性能比较

仍以上述 6 个 AS 上的路由表为对象进行分析，主要考察 AVL 数据结构的节点存储深度、搜索时间及占用的存储空间，并将分析结果与 PCT 数据结构进行比较。

(1)节点存储深度比较。从图 4 中可以看到，与 PCT 相比，AVL 的最大深度比 PCT 至少小 3，与理论值相差 0.7 左右。而且其最大深度和最小深度相差为 1，平均深度与理论值几乎吻合。克服了 PCT 的前缀节点分布不均匀、节点间深度差距大的缺点。不会因为节点的增加和删除而使 AVL 树的节点分布不均匀。

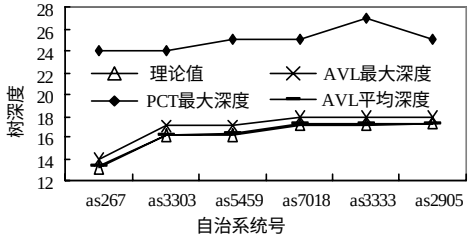

图 4　存储深度的比较　(下转第 9 页)