一种存储高效的IPv6路由查找方法

A Memory-Efficient IPv6 Route Lookup Approach

姜东虹/JIANG Donghong[1,2]，郑子豪/ZHENG Zihao[1,2]，李彦彪/LI Yanbiao[1,2]

（1. 中国科学院计算机网络信息中心，中国 北京 100083；
2. 中国科学院大学，中国 北京 100049）
(1. Computer Network Information Center, Chinese Academy of Sciences, Beijing 100083, China;
2. University of Chinese Academy of Sciences, Beijing 100049, China)

DOI：10.12142/ZTETJ.202406006
网络出版地址：http://kns.cnki.net/kcms/detail/34.1228.tn.20250108.1637.004.html
网络出版日期：2025-01-09
收稿日期：2024-10-15

摘要：由于IPv6前缀比IPv4更长，如何在保证查找性能的同时提高存储效率成为一个关键挑战。现有基于Trie、哈希和三态内容可寻址存储器（TCAM）的路由查找方法在存储效率和查找性能上均存在一定局限性。提出了一种基于前缀拆分模型的集合查找方法（SetSearch），旨在实现高效的IPv6路由查找与存储。SetSearch通过前缀拆分模型实现高效的片内存储，采用路由二维映射表，避免了叶推导致的IP前段爆炸问题，从而显著降低片外存储开销。此外，SetSearch还通过基于路由二维映射表的集合查找方法，将片外访存次数减少到最多一次。基于5个真实IPv6骨干路由器转发信息表（FIB）数据集的实验评估结果表明，SetSearch在片内存储、片外存储和片外访存次数等方面展现了优异的综合性能。

关键词：IPv6；路由查找；转发信息表；高效存储

Abstract: Given that IPv6 prefixes are longer than IPv4, enhancing storage efficiency without compromising lookup performance has become a critical challenge. Existing route lookup methods—such as Trie-based structures, hashing, and ternary content addressable memory (TCAM)—have limitations in storage efficiency or lookup speed. To address these challenges, this paper proposes SetSearch, a method based on the prefix-split model, to achieve efficient IPv6 route lookup and storage. SetSearch enhances on-chip storage through prefix splitting and uses a two-dimensional routing table to prevent the explosion of IP prefixes caused by leaf-pushing, significantly reducing off-chip storage demands. Furthermore, SetSearch minimizes off-chip memory accesses to a maximum of one by utilizing a set search strategy based on the two-dimensional routing table. Experimental evaluations using five real-world IPv6 backbone router forwarding information bases (FIBs) datasets demonstrate that SetSearch offers superior performance across metrics such as on-chip storage, off-chip storage, and off-chip memory access efficiency.

Keywords: IPv6; route lookup; forwarding information base; memory-efficient

引用格式：姜东虹，郑子豪，李彦彪．一种存储高效的IPv6路由查找方法[J]. 中兴通讯技术，2024, 30(6): 31-38. DOI: 10.12142/ZTETJ.202406006

Citation: JIANG D H, ZHENG Z H, LI Y B. A memory-efficient IPv6 route lookup approach [J]. ZTE technology journal, 2024, 30(6): 31-38. DOI: 10.12142/ZTETJ.202406006

路由查找是路由器、三层交换机等网络设备的核心功能，直接影响网络中数据包的转发性能。其主要任务是根据每个到达数据包的目的IP地址，在转发信息表（FIB）中找到最匹配的IP前缀，并依据与该前缀相关联的“下一跳”信息来转发数据包。随着无类别域间路由（CIDR）[1]技术的广泛应用，路由查找从简单的精确匹配问题演变为更复杂的最长前缀匹配（LPM）问题，成为网络设备中资源消耗最为显著的功能之一。

由于IPv4地址资源的枯竭，全球正在加速部署下一代互联网协议IPv6。然而，IPv6 FIB的高效存储面临着严峻挑战。一方面，由于IPv6前缀更长，相较于IPv4 FIB，IPv6 FIB在相同规模下需要消耗更多存储资源；另一方面，当前骨干路由器中的IPv6 FIB前缀数量已接近22万条[2]，并且仍呈指数增长。因此，研究一种存储高效的

基金项目：国家自然科学基金项目（62072430）

IPv6路由查找方法对于全球平稳过渡至下一代IPv6网络具有重要的现实意义。

当前的路由查找方法主要分为3类：1）基于三态内容可寻址存储器（TCAM）的方法[3-4]。该类方法的性能较高，因为TCAM能够在一个时钟周期内完成所有IP前缀的并行匹配。然而，这类方法存在功耗高且存储容量受限的缺点。另两类是基于算法的方法。2）基于哈希的方法[5-6]。哈希方法具有较小的存储开销，但哈希冲突可能导致最差情况下路由查找的访存次数无法确定。现有方法通常依赖多次哈希，这在硬件路由器中实现成本较高，因此基于多次哈希的方法更多用于软件路由器。3）基于特里树（Trie）的方法[7-8]。基于Trie的方法是目前业界主流的路由查找方法。通常，这类方法会基于芯片内存储器的流水线结构，即将Trie的不同层映射到流水线的不同流水级，不同流水级拥有独立的存储器，以实现高速流水线式并行查找。然而，尽管业界已经提出了多种Trie压缩技术[9]，但在应对更长的IPv6前缀时，基于Trie的IPv6路由查找方法的存储效率仍然较低。

为提高FIB的存储效率，一种基于前缀拆分的新型路由查找模型被提出[10]。该模型通过分析骨干路由器FIB中IP前缀间存在大量相似性，并通过一种前缀拆分与合并的方法来捕获这种相似性。具体来说，该模型通过选择一个拆分位置P，将长度小于等于*P*的IP前缀划分为FIB1，长度大于*P*的前缀划分为FIB2；随后，FIB2中的每条IP前缀以P为界被拆分为前半部分（称为IP前段）和后半部分（称为新IP前缀）；最终，相同的新IP前缀被合并形成FIB3，与每条合并后的新IP前缀关联的是一个IP前段集合。当前基于前缀拆分模型的路由查找方法（SplitTrie）[10]将FIB1中的前缀和FIB3中的新前缀分别映射到片内基于SRAM的流水线中，而FIB3中与新前缀关联的IP前段集合则映射到片外基于DRAM的大存储介质中。查找过程中，系统分别查找FIB1和FIB3，并取两者的最优值。由于FIB3中合并了大量的新IP前缀，片内存储开销显著降低。然而，当前基于前缀拆分模型的路由查找方法仍面临两大关键挑战，限制了其实际应用：一是，基于非叶推[11]的前缀拆分模型会导致单次路由查找的片外访存次数过多，严重影响查找性能；二是，基于叶推的前缀拆分模型会导致片外IP前段集合条目爆炸，以致片外存储开销巨大。

基于前缀拆分的路由查找模型[10]，我们提出了一种存储高效的IPv6路由查找新方法（SetSearch）。SetSearch在保留前缀拆分模型带来的片内存储高效优势的同时，还兼顾了查找性能和片外存储效率。

1 背景及相关工作介绍

本节首先以单比特特里树（Uni-bit Trie）为例，介绍当前业界主流的基于Trie的流水线化路由查找方法；接着，介绍前缀拆分模型并分析其优势；最后，结合基于非叶推和基于叶推的两个具体路由查找实例，分析当前基于前缀拆分模型的路由查找方法的局限性。

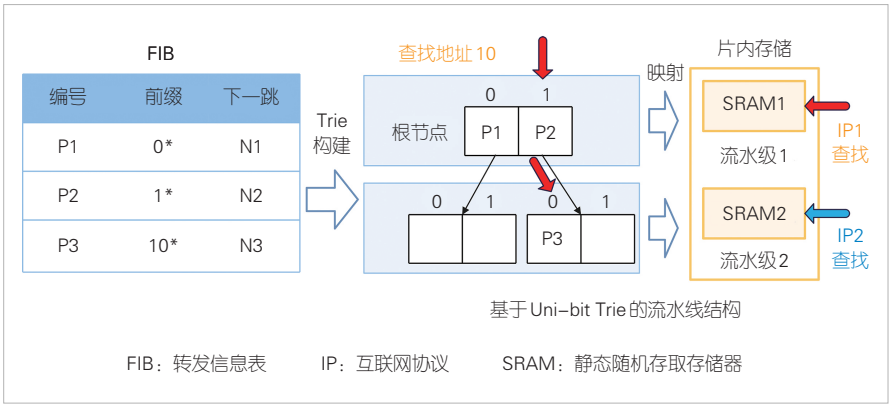
1.1 基于Uni-bit Trie的流水线式路由查找

基于Trie的路由查找方法通过将FIB组织成Trie结构，其中FIB中的每条IP前缀对应Trie中的一个实节点，且每条IP前缀由Trie根节点到其对应节点的路径表示。图1展示了如何将包含3条IP前缀的FIB构建成Uni-bit Trie结构，其中Trie节点由左右兄弟结构表示，即每个节点同时保存其兄弟节点的信息。Uni-bit Trie的查找过程从根节点开始，每次消耗IP地址的一位比特。如果该位为0，则跳转到当前节点的左孩子节点；若为1，则跳转到当前节点的右孩子节点。图1还展示了目的IP地址10的查找过程。

基于Trie的流水线式路由查找将Trie的每一层映射到流水线结构的不同流水级，每个流水级拥有独立的存储资源，因此可以实现所有流水级的并行查找。图1还展示了IP1在流水级1上查找Trie的第一层和IP2在流水级2上查找Trie的第二层在同一周期内并行执行的示例。

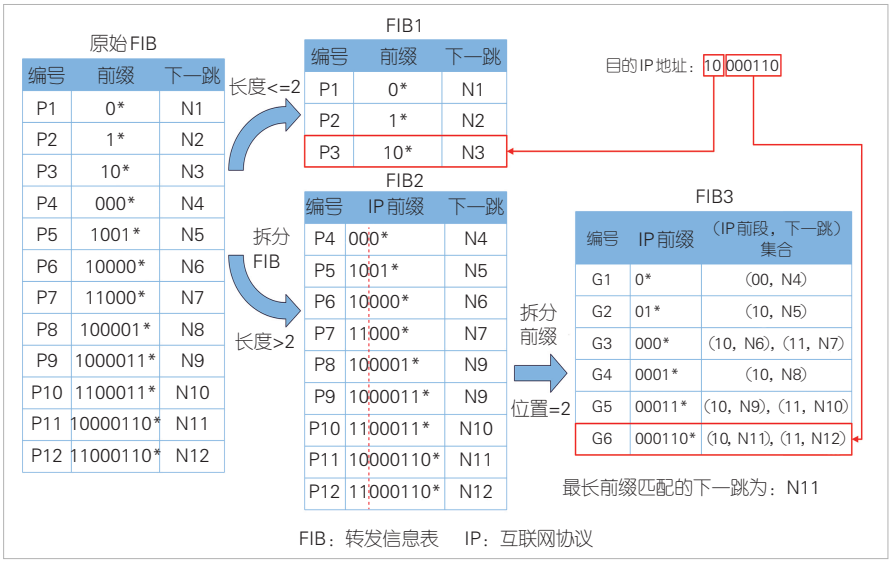
1.2 基于前缀拆分的路由查找模型及其优势分析

基于前缀拆分的路由查找模型（下文简称为前缀拆分模型）首先会选择一个拆分位置P，并依据选定的拆分位置将原始FIB拆分为FIB1和FIB2。其中，前缀长度小于等于*P*的



▲图1 基于Uni-bit Trie的流水线查找结构及IP查找示例

IP前缀被划分到FIB1中，长度大于*P*的IP前缀则划分到FIB2中。随后，以拆分位置P为界，将FIB2中的前缀进一步拆分为前半部分（IP前段）和后半部分（新IP前缀）。之后对相同的新IP前缀进行合并，形成FIB3。每条合并后的新IP前缀关联了一个IP前段集合。图2展示了对一个原始FIB进行前缀拆分的示例。在此示例中，拆分位置P选定为2，因此将长度小于等于2的0*、1*和10* 3条前缀划分到FIB1中，其余前缀划分到FIB2中。然后，依据拆分位置对FIB2中的前缀进行拆分与合并，形成FIB3。以P9和P10为例，这两条前缀的长度均大于2，因此被划分到FIB2中。拆分后P9和P10具有相同的后半部分，合并为同一条新IP前缀G5。G5关联的集合包含原P9、P10各自的IP前段及对应的下一跳信息。作为对比，FIB3中的IP前缀数目相比FIB2减少了3条。图2还展示了基于FIB1和FIB3的路由查找示例。

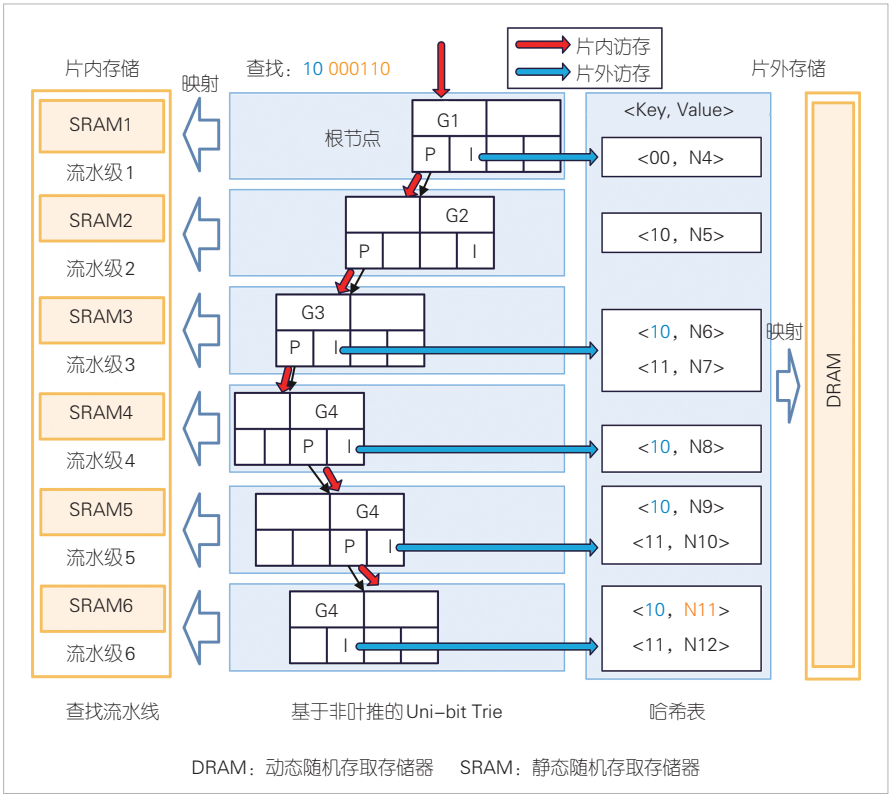


▲图2　前缀拆分过程及IP查找示例

前缀拆分模型在存储开销方面具有显著优势，原因在于经过合并后的FIB3中新IP前缀的数目相比FIB2大幅减少，能显著降低基于FIB3构建的Trie的存储开销。

1.3 当前基于前缀拆分模型的路由查找方法及其局限性

当前基于前缀拆分模型的路由查找方法SplitTrie[10]将FIB1和FIB3中的前缀分别构建为两棵不同的Trie，并分别映射到基于片内存储器（SRAM等）的流水线中。而FIB3中与新前缀关联的IP前段集合则映射到片外大容量存储介质中（DRAM等）。IP查找过程中，分别查找FIB1和FIB3，并取两者的最优值。图3展示了基于1.2节例子中的FIB3所构建的数据结构及其存储映射关系。



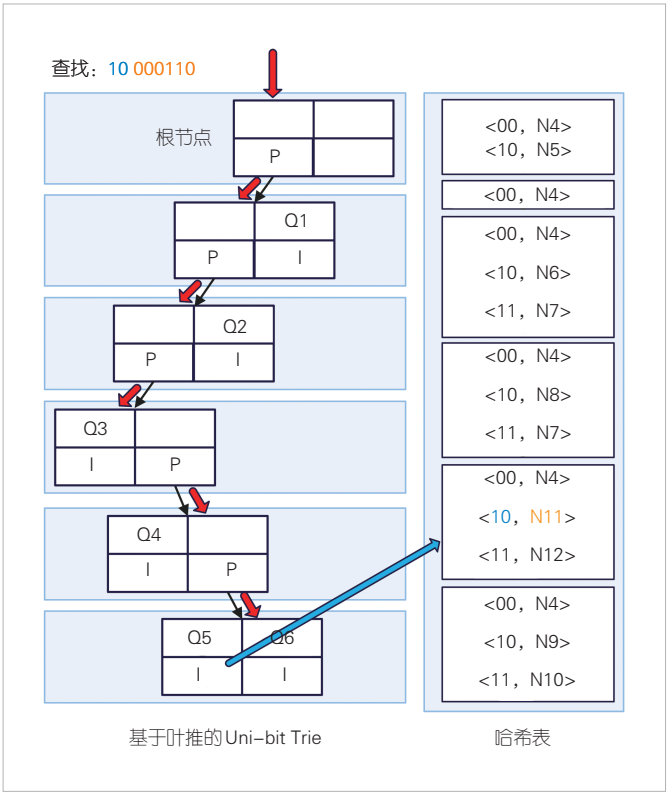
▲图3　基于非叶推的SplitTrie及IP查找示例

在IP查找过程中，SplitTrie针对FIB1的查找过程与1.1节中的例子相同。而针对FIB3进行查找时，SplitTrie首先依据拆分模型选定的拆分位置P，将IP地址拆分为前后两段。然后，使用后段在FIB3所构建的片内Trie中进行查找，每经过一个实节点（即表示匹配到一条IP前缀），则使用IP地址的前段在该IP前缀所对应的片外IP前段集合中进行查找（SplitTrie[10]使用了哈希表进行查找）。图3展示了IP地址10000110的查找过程。在该查找过程中，依次经过了G1、G3、G4、G5和G6 5个实节点，并进行了6次片外IP前段集合的查找，最终匹配到G6节点，查找结果为下一跳N11。

然而，SplitTrie将FIB3的IP前段集合（哈希表）存储在片外，因此在一次IP查找过程中，可能会进行多次片外访存。由于片外访存延迟较高，在实际应用中，单次IP查找通常只允许最多一次片外访存。过多的片外访存将显著降低IP查找性能。

为减少片外访存次数，可以引入经典的叶推技术[11]，即将所有中间实节点推送至叶子节点，使得片外访存仅发生在叶子节点，从而实现单次IP查找最多一次片外访存。图4展示了在图3示例的基础上引入叶推技术后的数据结构和相同IP地址的查找示例。在这一查找过程中，只经过叶子节点Q5一个实节点，并进行了仅一次片外访存。然而，基于叶推的SplitTrie进行叶推时，需要将FIB3中新IP前缀对应的IP前段集合一并推送至叶子节点，这会导致IP前段的大量复制，进而显著增加片外存储开销。例如，图4中叶推后的IP前段总数目由叶推前的9个增加到叶推后的15个。

综上所述，基于前缀拆分模型的非叶推SplitTrie虽然显著降低了片内存储开销，但多次片外访存严重影响了查找性能。基于叶推的SplitTrie虽然将单次IP查找的片外访存次数优化到最多一次，但叶推引发的IP前段爆炸问题会导致片外存储开销巨大。因此，当前基于前缀拆分模型的路由查找方法仍然面临严峻挑战，限制了其实际应用。



▲图4 基于叶推的SplitTrie及IP查找示例

2 基于前缀拆分模型的交集路由查找方法

为应对IPv6 FIB在存储开销方面所面临的挑战，本文中我们提出了一种存储高效的IPv6路由查找方法SetSearch。首先，SetSearch基于前缀拆分模型，在片内存储效率上表现出色；其次，利用基于路由二维映射表的集合查找方法，SetSearch在无需叶推的情况下即可实现单次IP查找仅需一次片外访存；最后，SetSearch在实现单次片外访存的同时还避免了叶推所带来的IP前段爆炸问题，相比于基于叶推的SplitTrie方法大幅降低了片外存储开销。

2.1 路由二维映射表

SetSearch是一种基于前缀拆分模型的方法。针对由前缀拆分模型生成的FIB1和FIB3，SetSearch采用不同的查找方法。首先，对于FIB1的查找，SetSearch同样使用基于片内存储器（SRAM等）的流水线式查找方法；对于FIB3的查找，SetSearch则使用了一种基于交集查找的新方法。由于针对FIB1的查找在1.1节已有详细描述，后文重点介绍针对FIB3的交集查找方法。

交集查找方法的第一步是将前缀拆分模型生成的FIB3拆解为IP前缀映射表和IP前段映射表。首先，提取FIB3中所有的新IP前缀及其对应的编号，构成IP前缀映射表。接下来，为FIB3中的每种IP前段找出所有包含该IP前段的IP前缀，并提取它们的编号以及对应的下一跳信息，生成(编号,下一跳)二元组集合。例如，在图5中的FIB3中，包含IP前段11的IP前缀包含000*、00011*和000110*，这些IP前缀的编号和

FIB3

编号	IP前缀	(IP前段，下一跳)
G1	0*	(00，N4)
G2	01*	(10，N5)
G3	000*	(10，N6)，(11，N7)
G4	0001*	(10，N8)
G5	00011*	(10，N9)，(11，N10)
G6	000110*	(10，N11)，(11，N12)

拆解

IP前缀映射表

IP前缀	编号	IP前缀	编号
0*	G1	0001*	G4
01*	G2	00011*	G5
000*	G3	000110*	G6

IP前段映射表

IP前段	(编号，下一跳)集合
00	(G1，N4)
10	(G2，N5)，(G3，N6)，(G4，N8)，(G5，N9)，(G6，N11)，
11	(G3，N7)，(G5，N10)，(G6，N12)

FIB：转发信息表 IP：互联网协议

▲图5 路由二维映射表示例

关联的下一跳二元组分别为(G3,N7)、(G5,N10)和(G6,N12)。所有IP前段和其对应的(编号,下一跳)二元组集合构成IP前段映射表。图5展示了由图2中FIB3生成的路由二维映射表示例。

2.2 数据结构及其存储映射

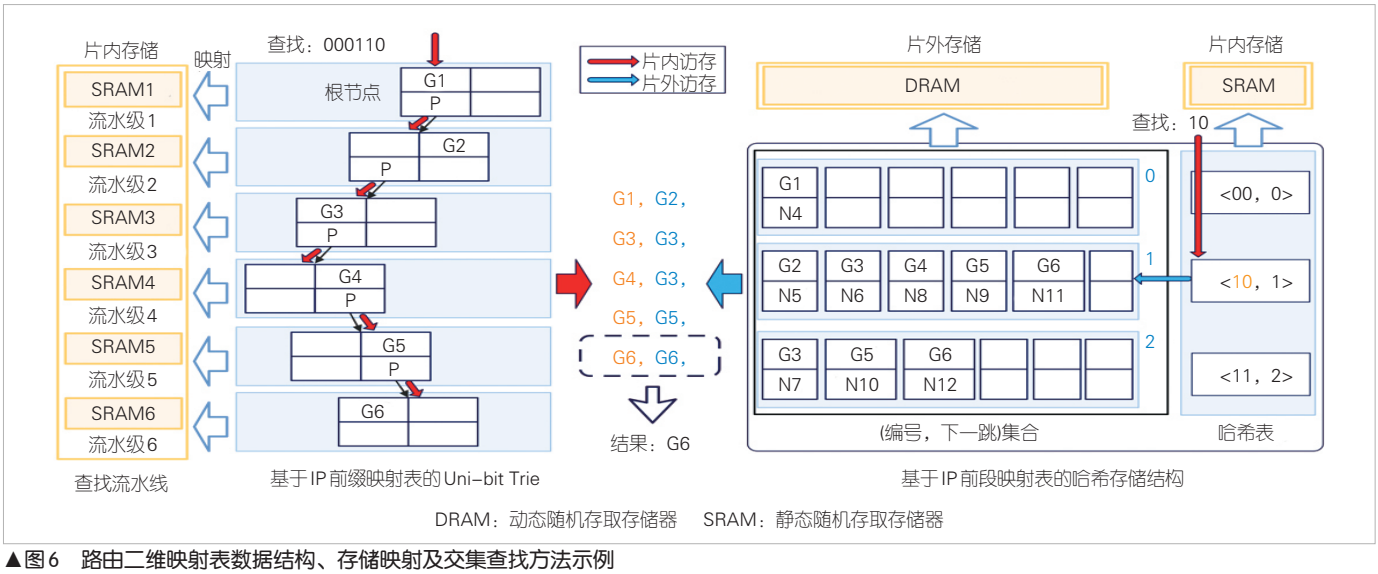
SetSearch针对FIB1的数据结构构建及其映射方式与1.1节一致。针对FIB3的交集查找方法的数据结构由两部分组成。第一部分是基于IP前缀映射表构建的非叶推Trie。该Trie的不同层被映射到流水线中的不同流水级，各流水级对应片内独立的存储器（SRAM等）。第二部分是基于IP前段映射表构建的哈希表，其中哈希表的键为IP前段，值为指向其关联的(编号,下一跳)二元组集合的索引值。哈希表被映射到片内独立存储器（SRAM等）中，而(编号,下一跳)二元组集合则被映射到片外存储器（DRAM等）中。图6展示了与图5中路由二维映射表示例相关的数据结构及其存储映射关系示例。

2.3 SetSearch查找方法

SetSearch针对FIB1的查找方法与1.1节一致。而针对FIB3的交集查找方法则分为两个阶段。第一阶段并行查找Trie和哈希表，得到两个查找结果集合；第二阶段计算这两个结果集合的交集，并选取交集中最长IP前缀对应的下一跳作为最终查找结果。如图6所示，假设待查找的IP地址为10000110，则取其后6位在映射到片内存储的Trie中进行查找，依次经过实节点G1、G3、G4、G5和G6，因此Trie的查找结果集合为{G1,G3,G4,G5,G6}。同时，取IP地址的前2位在哈希表中进行查找，命中哈希条目<10,1>，对应的(编号,下一跳)二元组集合索引值为1。进一步通过索引值从片外存储中获取查找结果集为{G2,G3,G4,G5,G6}。两个结果集的交集为{G3,G4,G5,G6}，其中G6对应的IP前缀最长，因此最终查找到的下一跳为与G6对应的下一跳N11。在这一查找过程中，系统共访问片内存储7次，其中Trie查找访问了6次，哈希表查找访问了1次；片外DRAM存储则访问1次，用于获取(编号,下一跳)二元组集合。SetSearch查找方法的完整查找过程伪代码如算法1所示。

算法1： SetSearch查找

输入：根节点 *root*, IP地址 *addr*;
输出：下一跳 *nh*.
① *SetSearch*(*root*, *nh*):
② $(addr_1, addr_2) \leftarrow$ 拆分 *addr*; // 拆分IP地址
③ $nh_1 \leftarrow root.Trie_1.lookup(addr_1)$; // 查找FIB1对应的 $Trie_1$
④ $InfoSet_1 \leftarrow root.Trie_3.lookup(addr_2)$; // 查找FIB3对应的 $Trie_3$
⑤ $index \leftarrow HashTable[addr_1]$; // 查找哈希表
⑥ $InfoSet_2 \leftarrow SetMap[index]$; // 获取二元组集合
⑦ $InfoSet \leftarrow InfoSet_1 \cap InfoSet_2$;
⑧ $nh_2 \leftarrow$ 取 *InfoSet* 中最长IP前缀的下一跳;
⑨ if $nh_1 \neq \varphi$ and $nh_2 \neq \varphi$:
⑩ $nh = nh_1.pfxlen < nh_2.pfxlen\ ?\ nh_2 : nh_1$;
⑪ else if $nh_1 = \varphi$ and $nh_2 = \varphi$:
⑫ $nh =$ 默认路由下一跳;
⑬ else:
⑭ $nh = (nh_1 == \varphi)\ ?\ nh_2 : nh_1$;
⑮ end if;
⑯ return *nh*;



▲图6　路由二维映射表数据结构、存储映射及交集查找方法示例

2.4 SetSearch查找复杂度分析

SetSearch查找方法的时间复杂度可以细分为以下几个部分。首先，IP地址拆分的时间复杂度为O(1)。其次，由于Trie查找的复杂度与地址长度线性相关，因此在$Trie_1$和$Trie_3$中进行前缀查找时，时间复杂度分别为O(|addr1|)和O(|addr2|)，其中|addr1|和|addr2|表示地址长度。哈希表的查找和SetMap的直接索引访问平均时间复杂度为O(1)。计算结果集InfoSet1和InfoSet2的交集操作，时间复杂度为O(|InfoSet1| + |InfoSet2|)。在交集InfoSet中选取最长前缀对应的下一跳，最坏情况下时间复杂度为O(|InfoSet|)。其余的条件判断和返回操作均为常数时间。因此，SetSearch方法的理论时间复杂度可表示为O(|addr1| + |addr2| + |InfoSet1| + |InfoSet2| + |InfoSet|)。然而，在实际应用中，考虑到地址长度和集合大小的限制，复杂度可以近似认为是常数级别，即O(1)。

3 实验评估与结果分析

3.1 实验设置

为了评估基于前缀拆分模型的集合查找方法（SetSearch），本文选取了以下3种方法进行对比：基于Uni-bit Trie的查找方法（Trie）、基于前缀拆分模型的Uni-bit Trie查找方法（SplitTrie）以及基于叶推的SplitTrie查找方法（SplitTrie-LP）。实验数据集来源于不同大洲的5个互联网交换中心（IXP）骨干路由器的真实IPv6 FIB数据[12]：RRC01、RRC11、RRC15、RRC19和RRC23。它们的IPv6前缀数目分别为203000、203411、210078、197051和203476。这些数据反映了2024年1月25日上午8:00的路由器状态。评估指标主要集中在片内存储开销、片外存储开销、片内访存次数以及片外访存次数。

3.2 参数选择实验

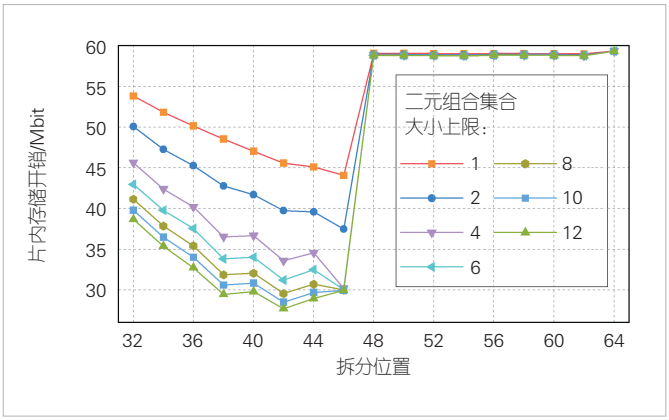
在SetSearch方法的实现过程中，片内存储开销主要受到两个参数的影响：拆分位置和与IP前段对应的(编号,下一跳)二元组集合的大小上限。其中，拆分位置决定了原始FIB中有多少IP前缀可以被拆分与合并；二元组集合大小上限决定了每个IP前段能够在片外存储的二元组数量，超出上限的二元组将导致原始IP前缀被重新划分给FIB1。

图7展示了拆分位置与二元组集合大小上限对SetSearch片内存储开销的影响。在实验中，每个二元组占用40 bit（编号和下一跳各20 bit），而单次片外访存的位宽上限设置为512 bit，因此二元组集合大小的理论最大值为12。从图7可以看出，在二元组集合大小上限小于等于6的情况下，拆分位置选择46时SetSearch的片内存储开销最小；而在二元组集合大小上限大于6的情况下，拆分位置选择42时SetSearch的片内存储开销最小。此外，图7还显示，当二元组集合大小上限为12、拆分位置为42时，SetSearch的片内存储开销在所有情况下最小。因此，后续实验中，SetSearch方法始终采用这一参数配置。

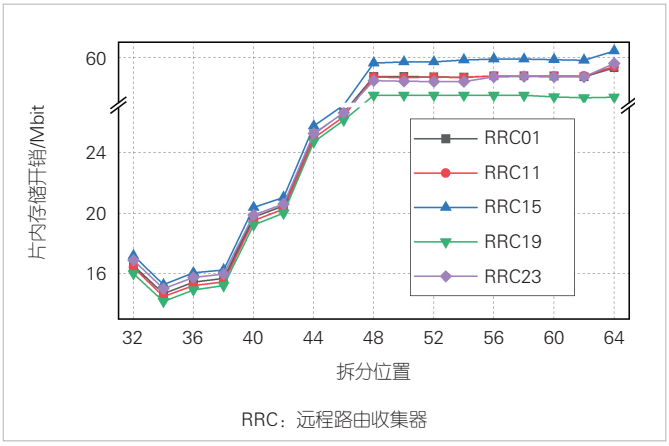
相比之下，SplitTrie方法的片内存储开销仅受拆分位置的影响。图8展示了拆分位置对SplitTrie片内存储开销的影响。从图中可以看出，在所有5个数据集中，拆分位置为34时，SplitTrie的片内存储开销最小。因此，后续实验中，SplitTrie和SplitTrie-LP方法均采用该参数设置。

3.3 对比实验

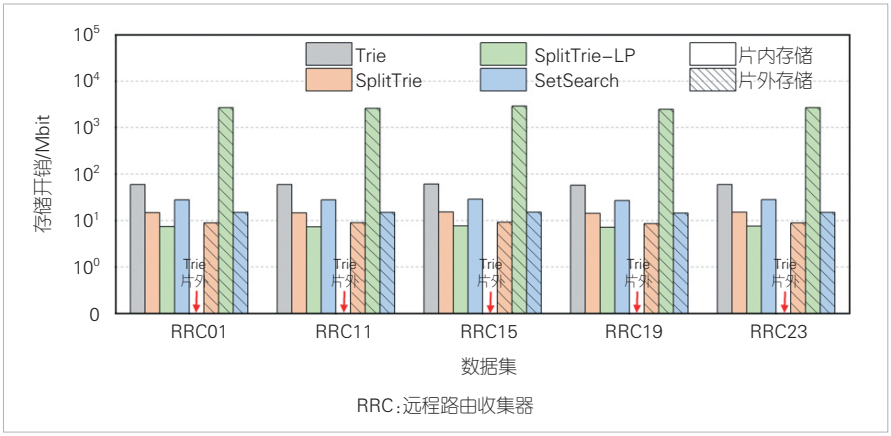
图9和图10分别展示了Trie方法、SplitTrie方法、SplitTrie-LP方法以及SetSearch方法的片内外存储开销和片内外访存次数。在存储开销方面，由于片内存储资源稀缺且昂贵，我们更关注片内存储开销；相比之下，片外存储资源更丰富且成本较低，可以适度容忍更高的片外存储开销，但



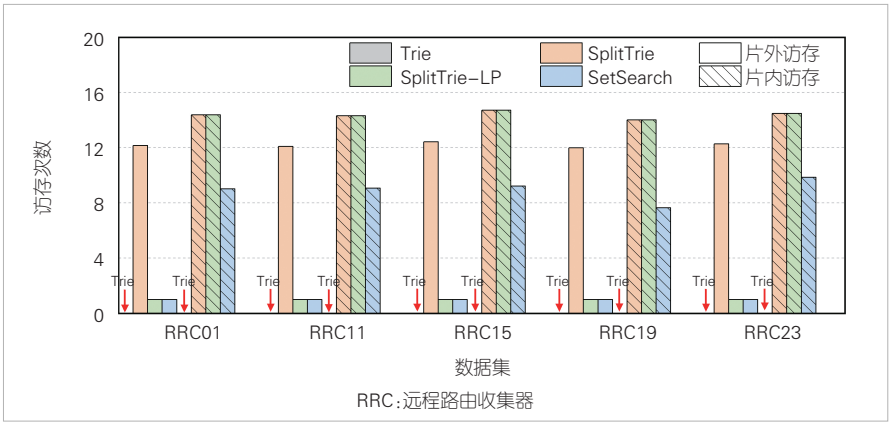
▲图7 拆分位置与二元组集合大小上限对SetSearch片内存储开销的影响



▲图8 拆分位置对SplitTrie片内存储开销的影响



▲图 9　存储开销对比



▲图 10　访存次数对比

不能过于庞大。因此，存储开销分别统计片内外的总和，即存储原始FIB的总存储开销。在访存方面，SetSearch和对比方法均采用流水线式并行查找，因此查找吞吐量仅受查找引擎主频影响，而与片内访存次数无关。然而，最大访存次数会影响查找时延。单次片内访存时延通常在纳秒级别，影响较小；但片外访存时延较高，一般的路由查找引擎最多只能容忍一次片外访存。此外，由于基于拆分模型的方法只关心FIB3的优化，因此本文只统计查找FIB3相关前缀的平均片内外访存次数。

SetSearch方法的平均片内存储开销为27.8 Mbit，平均片外存储开销为14.8 Mbit，平均片内访存次数为9次，平均片外访存次数为1次。从对比来看，虽然SetSearch方法在片内存储开销、片外存储开销以及片外访存次数等单个指标上并非最佳，但其综合表现最优，且各项性能均衡。具体来说，与Trie方法相比，SetSearch的片内存储开销平均下降了53.2%；与SplitTrie方法相比，SetSearch的片外访存次数下降了91.8%；与SplitTrie-LP方法相比，SetSearch的片外存储开销下降了99.4%。

尽管当前的SetSearch方法在综合表现方面具有较优性能，但在实现层面仍然面临一个关键挑战：如何降低IP前缀映射表和IP前段映射表查找结果的交集操作的开销。该问题的优化方向主要有两个：一是降低IP前缀映射表所构建的Trie深度，二是减小二元组集合的大小上限。

对于优化方向一，由于IPv6前缀主要集中于长度32和48之间，且3.2节中实验表明选择拆分位置为42较优，因此一种可行的方法是将FIB3中前缀长度范围限制为43至48，从而将IP前缀映射表所构建的Trie最大深度限制为8。对于优化方向二，一种可行的方法是通过选择关键比特位，将二元组集合分散到不同存储区域，从而进一步减少二元组结果集合。针对上述优化，我们将在未来的研究工作中继续探索。

4 结束语

针对IPv6路由查找中FIB存储开销大的问题，我们提出了一种基于前缀拆分模型的集合查找方法（SetSearch）。该方法基于前缀拆分模型，通过路由二维映射表的构建和集合查找机制，在不引入叶推操作的情况下，有效降低了片内和片外的存储开销，并实现了单次路由查找仅需一次片外访存。实验结果表明，与现有方法相比，SetSearch展现出了更为均衡的综合性能，在片内存储开销、片外存储开销和片外访存次数方面的综合表现最优。

参考文献

[1] FULLER V, LI T. Classless inter-domain routing (CIDR): the internet address assignment and aggregation plan [EB/OL]. [2024-10-25]. https://tools.ietf.org/html/rfc4632
[2] BGP. AS131072 IPv6 BGP table data [EB/OL]. [2024-10-25]. https://bgp.potaroo.net/v6/as2.0/index.html, 2024-09-11
[3] ZANE F, NARLIKAR G, BASU A. CoolCAMs: power-efficient TCAMs for forwarding engines[C]//IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No. 03CH37428). IEEE, 2003, 1: 42-52
[4] HE P, ZHANG W Y, GUAN H T, et al. Partial order theory for fast TCAM updates [J]. IEEE/ACM transactions on networking, 2018, 26(1): 217-230. DOI: 10.1109/TNET.2017.2776565

[5] WALDVOGEL M, VARGHESE G, TURNER J, et al. Scalable high speed IP routing lookups [C]//Proceedings of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication. ACM, 1997: 25–36. DOI: 10.1145/263105.263136
[6] JIANG D H, LI Y B, CHEN Y X, et al. Heuristic binary search: adaptive and fast IPv6 route lookup with incremental prefix updates [C]//Proceedings of IEEE/ACM Transactions on Networking. IEEE, Dec. 2024: 1–16. DOI: 10.1109/TNET.2024.3504244
[7] EATHERTON W, VARGHESE G, DITTIA Z. Tree bitmap: hardware/software IP lookups with incremental updates [J]. ACM SIGCOMM computer communication review, 2004, 34(2): 97–122. DOI: 10.1145/997150.997160
[8] YANG T, XIE G G, LI Y B, et al. Guarantee IP lookup performance with FIB explosion [C]//Proceedings of the 2014 ACM conference on SIGCOMM. ACM, 2014: 39–50. DOI: 10.1145/2619239.2626297
[9] ASAI H, OHARA Y. Poptrie: a compressed trie with population count for fast and scalable software IP routing table lookup [J]. ACM SIGCOMM computer communication review, 2015, 45(4): 57–70. DOI: 10.1145/2829988.2787474
[10] LI Y B, ZHANG D F, HUANG K, et al. A memory-efficient parallel routing lookup model with fast updates [J]. Computer communications, 2014, 38: 60–71. DOI: 10.1016/j.comcom.2013.10.005
[11] BIRMAN K P, SRINIVASAN V, VARGHESE G. Fast address lookups using controlled prefix expansion [J]. ACM transactions on computer systems, 1999, 17(1): 1–40. DOI: 10.1145/296502.296503
[12] RIS Docs. RIPE NCC route collectors [EB/OL]. [2024–10–25]. https://ris.ripe.net/docs/route-collectors/#bgp-timer-settings

作 者 简 介

姜东虹，中国科学院大学在读博士研究生；主要研究领域为IP查找算法、高性能路由器数据平面；发表论文3篇。

郑子豪，中国科学院大学在读硕士研究生；主要研究领域为云网络的资源管理、路由转发；发表论文1篇。

李彦彪，中国科学院计算机网络信息中心研究员；主要研究领域为高效路由转发、互联网基础资源安全与卫星互联网；主持国家重点研发计划项目、国家自然科学基金面上项目等科研项目10余项；曾获中国电子学会技术发明奖一等奖；发表论文40余篇。