DOI：10.7652/xjtuxb201302011

# 面向转发的双向高速报文解析结构

董永吉[1]，郭云飞[1,2]
（1.国家数字交换系统工程技术研究中心，450002，郑州；2.解放军理工大学指挥信息系统学院，210007，南京）

**摘要**：为进一步提升未来互联网体系结构的实验平台对转发报文高速灵活解析的能力，提出了一种面向转发的双向报文解析结构(BiPPAF)。该结构由以下2个方面构成：在协议的解析表达上，利用二叉trie树动态灵活的字符串组织方式，实现协议解析表达的可扩展；在协议解析的处理上，采用硬件流水线通过高速流水的查表方式，实现协议解析的高性能。BiPPAF结构通过为每个网络接口建立协议二叉trie树实现接口独立的协议解析能力，并利用节点映射算法来完成流水线和二叉trie树之间的关联，能够均衡各级流水线上二叉trie树的节点数目优化存储空间的使用。仿真实验表明，与Packet Parsing方法相比，BiPPAF结构的协议处理速率提升了31%，而资源占用降低了64%。

**关键词**：报文解析；二叉trie树；流水线；未来互联网
**中图分类号**：TN914.42 **文献标志码**：A **文章编号**：0253-987X(2013)02-0063-06

# A Parsing Architecture for Bidirectional Forwarding-Oriented High-Speed Packet

DONG Yongji[1], GUO Yunfei[1,2]
(1. National Digital Switching System Engineering and Technological Center, Zhengzhou 450002, China; 2. College of Command Information Systems, People's Liberation Army University of Science and Technology, Nanjing 210007, China)

**Abstract**: A parsing architecture for bidirectional forwarding packet (BiPPAF) is proposed to enhance the experimental platform ability of future Internet architecture with high-speed and flexible parsing forwarding packets, The proposed method is composed of two aspects: the dynamic string organization of a binary trie is used to express protocol parsing scalably on the parsing expression of the protocol; and the high-speed look-up table of a hardware pipeline is used to achieve high-performance protocol parsing on processing protocol parsing. A protocol binary trie is established for each network interface to provide independent interface protocol parsing, and a node mapping algorithm is used to establish the association between the pipeline and the binary trie and to optimize the use of storage space for a balanced binary trie in all pipeline stages. Simulated experiments and comparisons with the Packet Parsing show that the proposed method improves the protocol processing by 31%, and reduces the resource consumption by 64%.

**Keywords**: packet parsing; binary-trie; pipeline; future network

近年来，国内外已相继从新型网络体系结构、可重构技术等不同侧面对新型信息通信网络体系进行了积极的研究和探索[1-2]。然而，由于传统路由平台的封闭性，在现有的互联网上验证和部署新体系结

收稿日期：2012-04-11。 作者简介：董永吉(1983—)，男，博士生；郭云飞(联系人)，男，教授，博士生导师。 基金项目：国家"973计划"资助项目(2012CB315901，2012CB315905)；国家"863计划"资助项目(2011AA01A103)；国家科技支撑计划资助项目(2011BAH19B01)。
网络出版时间：2012-10-31 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20121031.1100.002.html

构、新协议与新技术非常困难,已经严重制约了互联网体系结构的发展和创新[3]。

任何对未来网络体系架构[4-5]的理论研究成果和设想都需要在路由转发实验平台上进行广泛和深入的验证[6]。在路由器转发中,解析报文头部并萃取转发信息是比查表更基础的部分。为了满足网络体系架构下新协议的解析需求,Kobiersky 等人使用可扩展标记语言在现场可编程门阵列(field programmable gate array,FPGA)上实现了一个可线速处理 20 Gb/s 链路速率的可重构协议解析状态机[7],但在一些复杂的应用场景下,该状态机的解析速率会有所下降;Kozanitis 等人结合三态内容寻址存储器(ternary content addressable memory,TCAM)和哈希散列混合查表,提出了一种 Kangaroo 结构[8],通过提取协议类型区分查表协议个数的步进,合理地将查表操作分配到 TCAM 或哈希散列中,进而一次查表最大可以解析 3 个协议,但由于 TCAM 器件的结构缺点,限制了该系统的可扩展性;Attig 等人基于 FPGA 设计了一种简单直观的报文解析描述语言(packet pasring,PP)[9],通过对协议的特征描述,可以离线编译成一个协议解析处理器,经测试达到 400 Gb/s 的处理能力,但是该处理器占用 FPGA 内部资源较大,导致通用性较差。

为了加快对可重构信息通信基础网络理论体系中新协议的验证,本文设计了一种面向路由转发的协议解析结构(bidirectional packet parsing architecture for forwarding,BiPPAF),以满足可重构信息通信基础网络体系研究实验平台对协议快速而灵活解析的需求。

## 1 BiPPAF 结构介绍

二叉 trie 树是一种用于快速检索的二叉树结构,具有构建灵活、空间占用小、查找速度快的特点,而双向流水线则利用硬件并行处理的特性,在相同存储空间的条件下,可达到 2 倍的处理速率。为灵活并快速地实现新协议的解析,本文提出了由转发协议二叉 trie 树(forwarding protocol trie,FP_trie)和双向流水线 2 部分组成的 BiPPAF 结构,通过结合两者的优势实现快速灵活的协议解析功能。BiPPAF 结构如图 1 所示。

在 BiPPAF 结构中,转发协议二叉 trie 树将传统的报文解析过程转换为二叉 trie 树,实现报文解析的灵活表示;双向流水结构则充分利用硬件并行处理的特性,采用线性流水的查找 FP_trie 树实现协议解析,达到高速的处理能力。

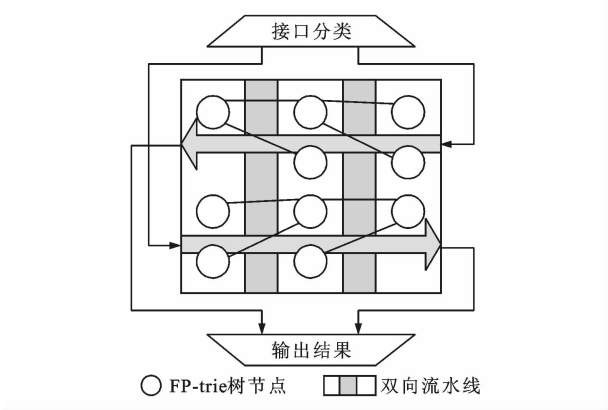



图 1 BiPPAF 结构组成图

### 1.1 转发协议二叉 trie 树

**定义 1** 节点深度是该节点到根节点的最大距离,根节点的深度为 1,其余节点的深度为双亲深度的最大值加 1;FP_trie 树中最大的节点深度称为 FP_trie 树的深度。

**定义 2** 叶子节点只能包含解析的结果,其中包含正常解析内容的称为实叶子节点,反之包含无法解析的称为虚叶子节点。

**定义 3** 若二叉 trie 树被称为 FP_trie 树,则应满足如下条件:

**条件 1** 每个协议的识别判定可能需要多个定位规则,但每个非叶子节点内只存储一个定位规则和 2 个跳转的指针,其中定位规则包含该协议相关字段的判定取值及判定方法;

**条件 2** 任一个非叶子节点必须有 2 个子节点,但不能同时以 2 个实叶子节点作子节点;

**条件 3** 一个 FP_trie 树最多只能有一个虚叶子节点;

**条件 4** 从根到叶子节点的路径上的所有节点(不包括叶子节点)包含的定位规则顺序连接表示报文解析的过程。

由上述的定义可知,协议二叉 trie 树具有如下的性质。

**性质 1** 任一个可用 FP_trie 树描述的协议可用 $N$ ($N<+\infty$)个非叶子节点来判别表示。

**证明** 用反证法证明,假设协议 $P$ 可以用 FP_trie 树的 $N$ 个非叶子节点表示,$N=+\infty$。由条件 1 可知,每个非叶子节点都存储一个协议相关字段的判定值,则该协议 $P$ 需要 $+\infty$ 个协议字段来表示判定,这与协议实现的原则相悖,故这种情况不可能出现。所以,FP_trie 树描述的协议可以被有限个节

点来判别表示。

**性质 2**　当 FP_trie 树的深度为 $L$ 时，节点数 $S_L$ 满足 $L \leqslant S_L \leqslant 2^{L-1}$ 约束。

**证明**　用数学归纳法证明。

(1)归纳基础：当 $L=1$ 时，得到 $2^{1-1}-1=1$，因为深度为 1 的 FP_trie 树有且只有 1 个根节点，所以命题成立。

(2)归纳假设：假设对所有的 $k(1 \leqslant k < L)$ 命题成立，即 $k \leqslant S_k \leqslant 2^k-1$，证明 $k=L$ 时命题亦成立。

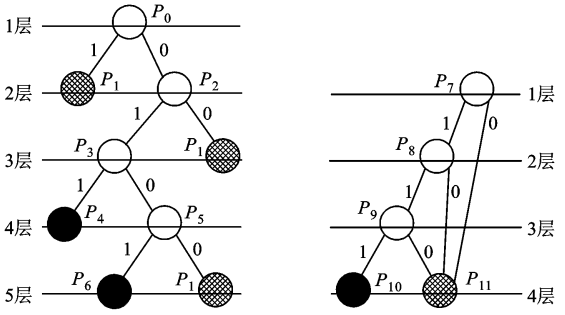
(3)归纳步骤：根据归纳假设，深度为 $L-1$ 的 FP_trie 树的节点数 $S_{L-1}$ 满足 $L-1 \leqslant S_{L-1} \leqslant 2^{L-1}-1$。由条件 2 可知，深度为 $L$ 的 FP_trie 树节点数至多比深度为 $L-1$ 的点数多 $2^{L-1}$ 个，故当 $k=L$ 时，深度为 $L$ 的 FP_trie 最多有节点 $2^{L-1}-1+2^{L-1}=2^L-1$ 个，满足 $S_L \leqslant 2^L-1$；由定义 1 可知，FP_trie 树深度为 $L$ 意味至少有一个节点的父节点的深度为 $L-1$，故 $L-1+1=L \leqslant S_L$。综上可知，当 $k=L$ 的时候，满足 $L \leqslant S_L \leqslant 2^{L-1}$，故命题成立。

**推论 1**　对于节点数 $S_L$ 一定的 FP_trie 树，深度满足 $\mathrm{lb}(S_L+1) \leqslant L \leqslant S_L$。

**证明**　由性质 2 可知，$L \leqslant S_L$ 且 $S_L \leqslant 2^L-1$，故 $S_L+1 \leqslant 2^L$，进而 $\mathrm{lb}(S_L+1) \leqslant L$，所以得到结论：$\mathrm{lb}(S_L+1) \leqslant L \leqslant S_L$。

综上可知，FP_trie 树的节点数和深度都存在上下限，即实现 FP_trie 树节点占用的存储空间有限，表明 FP_trie 树物理可实现。

目前，网络设备在多数情况下需要根据多协议标签交换协议(multi-protocol label switching，MPLS)和 IP 协议(Internet Protocol)进行数据的转发，本文根据网络实际情况，构造了 2 个协议树 A 与 B，其中协议树 A 描述了以太网协议内嵌虚拟局域网协议(virtual local area network，VLAN)及 MPLS、IPv6 的协议嵌套关系，而协议树 B 描述了 802.3SNAP 协议内嵌 IPv4 的协议嵌套关系，图 2 中分别对协议树 A 和协议树 B 进行了图形化表示。




(a)协议树 A　　(b)协议树 B

图 2　FP_trie 树结构

图 2 中的 2 个 FP_trie 树的节点类型共分为 3 种情况：①$P_1$ 和 $P_{11}$ 代表协议无法识别的叶子节点，用于指示报文无法解析；②$P_4$、$P_6$ 和 $P_{10}$ 分别代表了识别 MPLS、IPv4 和 IPv6 协议的叶子节点，用于输出协议解析的结果；③$P_0$、$P_2$、$P_3$、$P_5$、$P_7$、$P_8$ 和 $P_9$ 分别代表了 2 个协议树上的内部节点，用于协议的识别和判定，节点内容如表 1 所示。

表 1　FP_trie 树节点内容表

| 节点名称 | 判定协议 | 比较位宽/bit | 规则 | 判定内容 | 左偏移/bit | 右偏移/bit |
|---|---|---|---|---|---|---|
| $P_0$ | 以太网 | 16 | 大于 | 0x05DC | 0 | 0 |
| $P_2$ | VLAN | 16 | 等于 | 0x9100 | 32 | 0 |
| $P_3$ | MPLS | 16 | 等于 | 0x8847 | 0 | 0 |
| $P_5$ | IPv6 | 16 | 等于 | 0x08DD | 208 | 0 |
| $P_7$ | 802.3 | 16 | 大于 | 0x05DC | 16 | 0 |
| $P_8$ | 802.3 | 16 | 等于 | AAAA | 48 | 0 |
| $P_9$ | IPv4 | 16 | 等于 | 0x0800 | 144 | 0 |

## 1.2　BiPPAF 流水结构

BiPPAF 的双向流水结构用于存储 FP_trie 树的节点，并通过流水查找 FP_trie 树节点完成对协议解析的功能。通常 BiPPAF 由多级硬件流水线串联组成，而单级双向硬件流水线结构如图 3 所示。

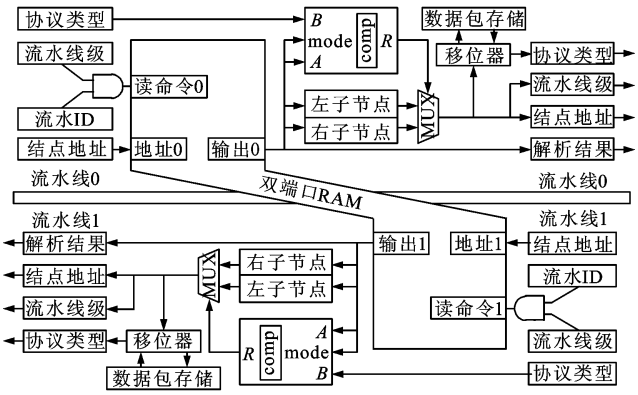



图 3　单级双向硬件流水线结构图

图 3 所示的单级双向硬件流水线以双口 RAM 为处理核心，配合比较器和移位器完成协议解析功能。当双向流水查找 FP_trie 树时，首先要匹配本级流水 ID，如果没有匹配，说明本级流水线上没有待查找到节点，所以跳过本级操作。其次，根据节点地址在双端口 RAM 中读取出对应的节点内容，并判断是内部结点还是叶子节点：如果是叶子节点，则

输出解析结果，且停止查找；如果是内部节点，则与协议类型在比较器中判别比较，选择匹配子节点输出，并根据数据偏移量在移位器中提取偏移后的数据。最后，将节点地址、流水线级和协议类型作为输出，送入相同处理方向的下级流水线。

### 1.3 BiPPAF 工作原理

为了实现网络虚拟化与隔离性对网络设备接口协议解析能力的动态需求，为每个虚拟路由接口提供独立的协议解析能力，并实现相互隔离的配置管理，BiPPAF 为每个接口都对应地建立一个 FP_trie 树，通过对每个单独的 FP_trie 树的管理，实现独立灵活的接口解析能力。硬件双向流水查表示意图如图 4 所示。

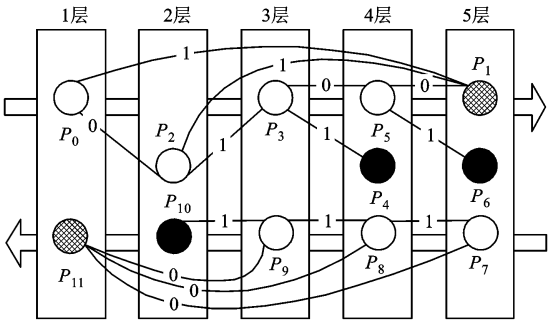



图 4 BiPPAF 流水查表示意图

图 4 中双向流水线上映射了 2 个 FP_trie 树，$P_0$ 和 $P_7$ 分别为 2 棵树的根节点。根据到达报文网络接口的不同，将报文送至对应解析树的根节点，并按照 FP_trie 树的映射方向进行解析查找。

## 2 节点映射算法

BiPPAF 结构构建过程中，需要将 FP_trie 树节点映射到双向流水线上，由于 FP_trie 树的结构不规则，理论上深度低的节点数量相对于深度高的节点数要少，所以不能简单的按照 trie 树的深度将节点顺序分配到硬件流水级上，虽然这种方式可以保证流水线每个时钟周期的查找请求，但是这种简单的映射方法会导致映射后的流水线存储空间使用不均衡，降低存储空间的利用率。针对 FP_trie 树映射过程中存储空间的均衡问题，提出双向映射的最优化数学模型如下

$$\min \max_{i=1,2,\cdots,H} G_i \tag{1}$$

式中：$H$ 表示流水线的深度；$G_i$ 表示第 $i$ 层流水线上的节点数。

**约束 1** 在单个映射方向上，每个父节点必须映射在比子节点距根节点更近的流水线级上。

**约束 2** 所有 FP_trie 树节点都唯一地映射在流水线上

$$\sum_{i=1}^{H} G_i = \sum_{j=1}^{n} M_j \tag{2}$$

式中：$n$ 表示接口对应 FP_trie 树的数量；$M_j$ 表示第 $j$ 个接口对应 FP_trie 树的节点数。

**约束 3** FP_trie 树的根节点只能映射在映射方向上流水线级的第一级。

### 2.1 双向映射算法

针对节点映射过程中流水线上节点存储分配不均衡的问题，本文提出一种双向映射算法（bidirectional mapping algorithm，Bi_MA），该算法分 2 个方向在流水线上映射网络接口 FP_trie 树的节点，通过正、反 2 个方向上 FP_trie 树映射节点数量的差异互补，实现流水线上节点的均衡存储。

Bi_MA 算法首先要将 FP_trie 树进行初始化处理，将待映射的 FP_trie 树集合中的元素分配到正向队列 $Q_F$ 和反向队列 $Q_R$ 中，其中 $Q_F$ 和 $Q_R$ 分别存储正、反 2 个方向上的 FP_trie 树，具体初始化流程如下：

(1)检查是否所有的 FP_trie 树都被分配到 $Q_F$ 和 $Q_R$ 中，若未全部分配，则执行步骤 2；否则，完成初始化；

(2)分别计算正向队列 $Q_F$ 中的节点总数 $N_F$ 和反向队列 $Q_R$ 中的节点总数 $N_R$，执行步骤 3；

(3)判断 $N_F$ 与 $N_R$ 的大小，若 $N_F > N_R$ 则将当前未分配的 FP_trie 树分配到 $N_R$ 中，否则将其分配到 $N_F$ 中，结束后返回步骤 1。

在将接口的 FP_trie 树初始化分配给队列 $Q_F$ 和 $Q_R$ 的同时，根据全部 FP_trie 树节点总数和硬件流水线的流水级数计算获得向上取整后的每级流水线节点个数平均值，然后按照每级流水线映射的节点个数不超过该平均值的规则，依次轮询队列 $Q_F$ 和 $Q_R$，将 FP_trie 树节点按照正、反 2 个方向交叉的映射到流水线中，具体流程如下所示：

(1)检查 2 个队列 $Q_F$ 和 $Q_R$ 中所有的 FP_trie 树是否都被映射到流水线上，若未全部映射，则执行步骤 2；否则，完成 Bi_MA 算法；

(2)检查正向队列 $Q_F$ 是否为空，若为空则执行步骤 3，否则执行步骤 4；

(3)检查反向队列 $Q_R$ 是否为空，若为空则执行步骤 1，否则转到步骤 5；

(4)在正向队列 $Q_F$ 中提取一个 FP_trie 树，按照先父节点再子节点的顺序，依次将节点按照流水

线映射方向依次对应存放到流水线级上，同时队列中待映射协议树的个数减 1，完成后返回步骤 3；

(5)在反向队列 $Q_R$ 中提取一个 FP_trie 树，类似于步骤 4 对 FP_trie 树节点的映射处理，在完成处理后返回步骤 2。

### 2.2　节点更新方法

为了快速更新流水线上 FP_trie 树节点，BiPPAF 采用了一种在流水线作业中通过插入 Write Bubble 的方法[10]实现快速更新。首先通过离线算法计算获得节点中更新的内容，然后利用流水线工作空闲进行实时的更新操作，具体更新方法如图 5 所示。在更新过程中，首先根据流水级 ID 找到匹配的流水线，然后根据节点地址找到需要更新的节点，利用正常协议解析流水查找的空隙，按照读/写指示在存储空间上更新节点。

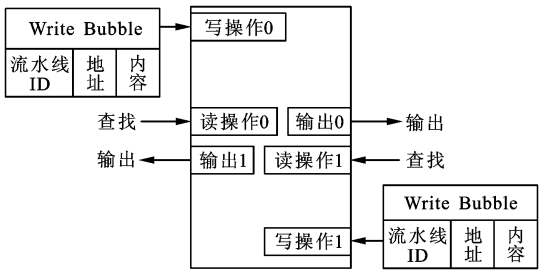



图 5　Write Bubble 节点更新方法

## 3　实验结果与分析

实验采用 NetFPGA-10G 网络系统开发平台，在平台的 Virtex-5 FPGA 内部模拟搭建了一个 32 路接口的 BiPPAF 结构，并模拟构造产生了 32 个接口的 FP_trie 树，所有 FP_trie 树的最大深度为 8，最小深度为 1，共有 396 个节点，其中实叶子节点 85 个，虚叶子节点 32 个。

采用本文 Bi_MA 算法和 simple 方法比较了映射后的流水线节点数分布，其中 simple 方法采用直接映射的方式，将 FP_trie 树节点按照其深度对应映射到相同深度的流水线级。仿真实验结果如图 6 所示。通过对比可以发现，由于 FP_trie 树形状具有不规则性，simple 方法映射后的流水线级上的节点分布极不均匀，不利于高效的硬件流水操作，而 Bi_MA 算法映射到流水线级的节点在流水线上均匀分布。相对于 simple 方法，Bi_MA 算法可以有效解决节点占用存储资源分布不均的问题，并充分优化使用存储空间。

表 2 给出在解析图 2 中的协议树 A 时，BiPPAF 与 PP、Kangaroo 和 BiPPAF-2 这 3 种协议解

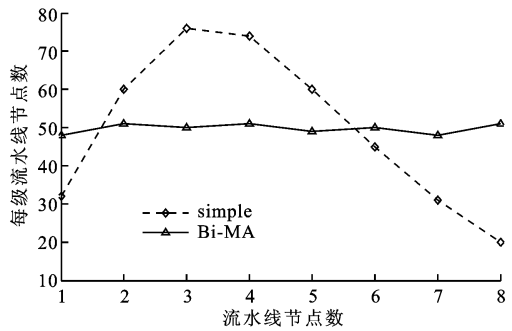



图 6　流水线节点数对比图

析结构在实验平台上的资源消耗和性能对比，其中 BiPPAF 的资源逻辑、BRAM 占用和时钟频率来源于 ISE 工具的布局布线报告，吞吐量则是根据 64 B 的以太网报文计算获得，由于 BiPPAF 结构的时钟频率为 218.77 MHz，并采用双通道流水查表，所以吞吐量可以达到 218.77 MHz×64×8 bit×2＝224 Gb/s。在资源占用上，相对于 PP，BiPPAF 虽然占用了少量的 BRAM 资源，但节省了近 32%的逻辑资源；而相对于 Kangaroo 算法，BiPPAF 在逻辑资源的占用上相差不大，但是节约了 10%左右的 BRAM 资源。在处理能力上，BiPPAF 的吞吐量远高于 Kangaroo 的 10 Gb/s，略低于 PP，但 BiPPAF 结构能够为每个接口提供专用的 FP_trie 树，所以不同于其他 2 种结构，可以为每个接口提供独立的解析能力。同时从资源占用上考虑，2 个并行的 BiPPAF 结构(表 2 中的 BiPPAF-2)占用的资源仍仅为 FPGA 逻辑资源的 14.7%，仅为 PP 占用的 36%，降低了 64%的逻辑资源占用，但吞吐量却可以达到 224 Gb/s×2＝448 Gb/s，相对于 PP 提升了近 31%的协议处理速率。因此，相对于已有的 2 种结构，BiPPAF 结构能在占用较少资源的情况下，提供较强的报文解析能力。

表 2　解析结构资源和性能对比表

| 结构 | 时钟频率/ns | 逻辑资源占用率/% | BRAM 占用率/% | 吞吐量/Gb·s$^{-1}$ |
|---|---|---|---|---|
| BiPPAF | 4.571 | 7.4 | 2 | 224 |
| PP | 2.999 | 40.2 | | 341 |
| Kangaroo | 15.873 | 6.4 | 15 | 10 |
| BiPPAF-2 | 4.571 | 14.7 | 4 | 448 |

## 4　结　论

本文结合双向流水线设计和二叉 trie 树查表思想，提出了一种应用于路由转发的双向报文协议解析结构 BiPPAF。通过构建 FP_trie 树来支持报文

协议解析的灵活度，采用硬件多级流水查表满足报文协议解析的高速处理，使用节点映射算法 Bi-MA 解决协议二叉 trie 树节点到流水线映射过程中存储资源不均衡的问题，并基于 NetFPGA-10G 实验平台仿真验证了 BiPPAF 的可行性和有效性，满足了可重构信息通信基础网络体系研究实验平台对协议解析能力的需求。

**参考文献：**


[1] PAUL S, PAN J, JAIN R. Architectures for the future networks and the next generation Internet: a survey [J]. Computer Communications, 2011, 34(1): 2-42.

[2] PAN Jianli, PAUL S, JAIN R, et al. A survey of the research on future Internet architectures [J]. IEEE Communications Magazine, 2011, 49(7): 26-36.

[3] PALKOPOULOU E, SCHUPKE D, BAUSCHERT T. Shared backup router resources: realizing virtualized network resilience [J]. IEEE Communications Magazine, 2011, 49(5): 140-146.

[4] 蒋林涛. 未来互联网的承载网络 [J]. 中兴通讯技术，2010，16(2)：10-12.
JIANG Lintao. The bearer network of the future Internet [J]. ZTE Communications, 2010, 16(2): 10-12.

[5] 林闯，贾子骁，孟坤. 自适应的未来网络体系架构 [J]. 计算机学报，2012，35(6)：1077-1092.
LIN Chuang, JIA Zixiao, MENG Kun. Research on adaptive future Internet architecture [J]. Chinese Journal of Computers, 2012, 35(6): 1077-1092.

[6] XIE Gaogang, HE Pen, GUAN Hongtao, et al. PEARL: a programmable virtual router platform [J]. IEEE Communications Magazine, 2011, 49(7): 71-77.

[7] KOBIERSKY P, KORENEK J, POLCAK L. Packet header analysis and field extraction for multi-gigabit networks [C]// Proceedings of the IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems. Piscataway, NJ, USA: IEEE, 2009: 96-101.

[8] KOZANITIS C, HUBER J, SINGH S, et al. Leaping multiple headers in a single bound: wire-speed parsing using the Kangaroo system [C]// Proceedings of the 29th IEEE Conference on Computer Communications. Piscataway, NJ, USA: IEEE, 2010: 830-838.

[9] ATTIG M, BREBNER G. 400 Gb/s programmable packet parsing on a single FPGA [C]// Proceedings of the ACM/IEEE Symposium on Architectures for Networking and Communication Systems. Piscataway, NJ, USA: IEEE, 2011: 12-23.

[10] BASU A, NARLIKAR G. Fast incremental updates for pipelined forwarding engines [C]// Proceedings of IEEE INFOCOM Conference. Piscataway, NJ, USA: IEEE, 2003: 64-74.


（编辑 刘杨）

（上接第 6 页）


[5] ARKIN A, ASKARY S, FORDIN S, et al. Web service choreography interface(WSCI) 1.0 [EB/OL]. [2012-05-06]. http://www.w3.org/TR/wsci/.

[6] BMI Working Group. Business process modeling language (BPML) [EB/OL]. [2012-05-06]. http://www.bpmi.org/.

[7] Web Services Choreography Working Group. WS-choreography definition language (WS-CDL) [EB/OL]. [2011-08-01]. http://www.ebpml.org/ws_-_cdl.htm.

[8] ABITEBOUL S, BENJELLOUN O, MANOLESCU I, et al. Active XML: peer-to-peer data and Web services integration [C]// Proc of the 28th Int'l Conf on Very Large Data Bases. Berlin, Germany: Morgan Kaufmann Publisher Inc., 2002: 1087-1090.

[9] SZALAY A, GRAY J. The world-wide telescope [J]. Communications of the ACM, 2002, 45(11): 50-54.

[10] MARTIN D, BURSTEIN M, HOBBS J, et al. OWL-S: semantic markup for web services [EB/OL]. [2012-05-06]. http://www.w3.org/Submission/OWL-S/.

[11] O'REILLY T. What is Web2.0 [EB/OL]. [2012-05-01]. http://www.oreillynet.com/pub/a/oreillyim/news/2005/09/30/what is-web-20.html.

[12] LERNER R M. At the forge-JavaScript, Forms and Ajax [J/OL]. Linux Journal, 2006(150) [2012-03-20]. http://www.linuxjournal.com/article/9160.

[13] RUSSELL A, SCHIEMANN D, SCHONTZLER D, et al. DOJO [EB/OL]. [2012-02-06]. http://dojotoolkit.org/.

[14] WANG Ge, COOK P R. On-the-fly programming: using code as an expressive musical instrument [C/OL]// Proceedings of the 2004 International Conference on New Interfaces for Musical Expression (NIME'04). [2011-08-12]. http://www.nime.org/proceedings/2004/nime2004_138.pdf.


（编辑 葛赵青 武红江）