

文章编号:1001-9081(2010)02-0306-03

基于网络处理器的并行包分类方法

刘震宇,李卫军,赖 粤

(华南理工大学 电子与信息工程学院,广州 510641)

(zhenyu.liu@163.com)

摘 要:在大型网络中大量的规则数量会导致位向量(BV)算法的位向量过长和稀疏,要在网络处理器中实现 BV 算法需要大量的存储资源,而且多次存储读取也降低了算法匹配效率。针对 BV 算法位向量的问题,将 Tuple 空间分割思想与 BV 算法相结合缩短了位向量长度,充分利用网络处理器的并行处理机制和硬件加速单元,提出了一种适用于网络处理器的改进算法——Tuple-BV 算法。该算法的元组分割缩短了位向量的长度,减少了位向量的存储空间和读取次数。通过对数据包处理延时的实验比较,当较多规则时,Tuple-BV 算法在最大延时和平均延时指标上优于 BV 算法。

关键词:网络处理器;位向量;元组空间;哈希表;Trie

中图分类号: TP393.01; TN918.9 **文献标志码:** A

Paralleled packet classification based on network processor

LIU Zhen-yu, LI Wei-jun, LAI Yue

(School of Electronic and Information Engineering, South China University of Technology, Guangzhou Guangdong 510641, China)

Abstract: There are a large number of rules in large networks. This feature leads to overlong and sparse bit vectors of BV algorithm, which requires a large amount of storage resources and too many read cycles in network processor. An improved method, Tuple-BV algorithm, was put forward, which combined tuple space partition with the BV algorithm to reduce the length of bit vector and take advantage of parallel processing mechanism and hardware acceleration units in network processor. The improved algorithm shortens the length of bit vector and reduces storage space. The experiments prove that Tuple-BV algorithm is better than BV algorithm in terms of maximum delay and average delay when the number of rules is over one thousand.

Key words: network processor; Bit Vector (BV); tuple space; Hash table; Trie

0 引言

对于 N 条规则,普通的并行算法在预处理过程需要为每一维规则建立一个 $(2N+1) \times N$ 的数组,随着规则数 N 的增长所需内存空间将指数急剧上升。位向量(Bit Vector, BV)算法作为一种改进的并行算法,通过加入 Trie 树结构后只需为规则的每一维建立一个 N 位的位向量,减少了内存空间的需求。但在大型网络中,规则的数目 N 会非常大,导致位向量相当长,需要对内存多次访问才能完整读取一个位向量,因此影响了 BV 算法的效率^[1-5]。

网络处理器是用来执行数据处理、转发的高速可编程处理器。网络处理器由于灵活性强等特点,给网络设备应用以及网络平台构建带来了极大的方便,被认为是下一代网络的核心技术之一^[4-7]。

由于网络处理器的快速存储资源有限,所以在网络处理器中实现 BV 算法,需要合理地使用存储资源,尽量减少对位图向量的读取。

本文将 Tuple 组元分割的思想引入到 BV 算法进行改进,提出了一种基于网络处理器的 Tuple-BV 算法,通过分析表明该算法利用元组分割减少了位向量的长度,从而减少位向量的存储空间和读取次数,提高了匹配效率。

1 网络处理器

IXP 系列的网络处理器由于其强大的灵活性一直受到各方面研究的重视。其中,IXP2850 网络处理器是为高端核心设备设计,其性能非常强大,具有 16 个 1.4 GHz 的微引擎,每个微引擎有 8 个硬件线程,支持 10 Gbps 应用,并且有一个 700 MHz 的处理核心 Xscale^[6-7]。

Scratch、SRAM 和 DRAM 构成了网络处理器的三大存储资源,如表 1^[6]。

表 1 网络处理器中主要存储资源比较

存储器	实际应用 空间/KB	片内 存储	读速度/ 指令周期	写速度/ 指令周期
Scratch Pad	16	是	100	40
SRAM	32	否	130	53
DRAM	704	否	295	53

表 1 中,Scratchpad 速度最快,但仅为 16 KB,常用于重要数据结构的缓存。SRAM 的速度与 Scratch 接近,容量较大,但是 SRAM 中的许多资源被系统占用,例如路由表占用了约 6 MB 的 SRAM 空间。DRAM 的容量最大,但读取速度最慢。

收稿日期:2009-08-03。 基金项目:教育部高等学校科技创新工程重大项目(707047)。

作者简介:刘震宇(1976-),男,湖南宁乡人,博士,主要研究方向:智能信号与信息处理、网络安全; 李卫军(1974-),男,广东韶关人,博士,主要研究方向:智能信息处理、移动抗干扰技术; 赖粤(1983-),男,广东梅县人,博士,主要研究方向:信息安全处理。

2 BV 算法及问题

2.1 BV 算法

设规则集合总共有 N 条规则,每条规则有 d 个域,将规则按照优先级顺序排列,记为 $R_k, 1 \leq k \leq N$,将所有规则映射到 d 维空间中的超矩形中。所有超矩形在每一维的边界将该维空间分割成不超过 $2N+1$ 个区间,区间记为 S_{ij} ,其中 i 表示维数, j 表示分割数, $1 \leq i \leq d, 1 \leq j \leq 2N+1$ 。对每个 S_{ij} 建立 N 长度比特的位向量 BV_{ij} ,如果规则 R_k 的第 i 个域值范围与 S_{ij} 有交集,则 BV_{ij} 的第 k 比特设置为 1,否则为 0。将上述建立的 S_{ij} 和 BV_{ij} 按照每一维 i 分别存储,建立匹配引擎,分割数 j 的大小按照规则分割的结果设置。

将 IP 包每维的值 $P_i (1 \leq i \leq d)$,分别输入到第 i 个域对应的引擎中并行匹配,每个匹配引擎返回 P_i 所在分割区间 S_{ij} 对应的 BV_{ij} ,最后将所有 BV_{ij} 进行“与”计算得到最终匹配的位向量 BV 。若 BV 的第 k 比特为 1,就表示 IP 包与规则 R_k 匹配。由于规则是按照优先级大小顺序排列的,即 BV 矢量中第一个比特 1 的位置对应的规则就是并行 BV 算法输出的与 IP 包匹配的规则。

如表 2 的规则集合,对于一个数据包为 (0011, 1010, 0111),通过对三个维度 Trie 树进行查找,依次得向量为: [000100101], [000000100], [000000100], 相与得 [000000100],所以匹配规则为 R7。

表 2 BV 算法匹配示例

规则索引	规则描述	位向量
R1	(01 *, 1 *, 11 *)	10000000
R2	(1 *, 11 *, 00 *)	01000000
R3	(1 *, 0 *, 10 *)	00100000
R4	(0 *, 1 *, 00 *)	00010000
R5	(10 *, 0 *, 10 *)	00001000
R6	(1 *, 11 *, 11 *)	00000100
R7	(0 *, 10 *, 01 *)	00000010
R8	(1 *, 1 *, 00 *)	00000001
R9	(0 *, 00 *, 11 *)	00000001

并行 BV 算法的优点是各个域的匹配过程可以并行处理,大大提高了速度^[1-2, 8]。

2.2 存在问题

在实际应用中,集中于一个匹配规则通常很少,即规则之间交叠的可能性不多。这个性质使得并行 BV 算法产生的每个位向量中比特 1 的个数是稀疏的。但是,每次匹配过程都要读取 d 个位向量相与,由于每个位向量的长度是 N ,导致 BV 算法需要访问多次内存才能读取一个位向量,使得 BV 算法的效率比较低^[3-5]。

如果要将 BV 算法在网络处理器上实现,由表 1 可见,网络处理器存储器的延时对读取位向量的影响非常大,过长的位向量会严重影响 BV 算法的执行效率。

3 Tuple-BV 算法

3.1 改进思路

从上一章分析可以知道,要在网络处理器上提高 BV 算法的效率,必须缩短位向量的长度,这样可以减少位向量的读取次数,降低存储器读取时延的影响。

在 Tuple 算法^[8]中,一个数据包与规则集合中的所有规

则的匹配操作可以转化为多个互不相交的、较简单的精确匹配操作,再从多个结果中选出最优匹配,这样优化了匹配性并简化了分组分类器的设计,也不需要增加存储空间。但是当一套规则集合转换成 Tuple 规则集合时, Tuple 规则的数目通常不可确定,即并行处理数目不可确定,这给并行处理带来了困难。

结合以上两种算法的特点,可以利用 Tuple 对 BV 算法的位向量进行改进,减少位向量中比特 0 的冗余,根据这个思路,提出 Tuple-BV 算法。 Tuple-BV 算法包括预处理和匹配两个过程。

3.2 预处理

假设规则集合有 N 条规则,每条规则有 d 维,一般规则考虑五元组匹配, $d \leq 5$,规则集合的预处理过程如下:

1) 对规则集合中所有的维度统一为前缀表示,例如,一般规则集合中的端口字段都是以范围表示的,需要将其转换为前缀表示,一般使用的方法是范围嵌套等级。

2) 将规则集合进行 Tuple 空间划分,形成多个不相重复的规则集合。假设维度 $d = 5$,那么: $T[2, 3, 2, 1, 3]$ 表示第一至第五维的前缀个数分别为 2, 3, 2, 1, 3 的规则集合,例如 $R = (11 *, 011 *, 10 *, 0 *, 000 *)$ 就属于 T 。

3) 为规则集合的每一个维度构建一个二进制 Trie 结构,即对于 d 维匹配来说需要构造 d 个二进制 Trie 结构。该二进制 Trie 结构每个节点最多包含 2 个指针,分别指向左右子节点,代表前缀的某个比特位为“0”和“1”时两种情况的搜索分支,每个节点存储着一个 Tuple 位向量,位图的位数为 m 。根据规则的第 i 维分量在第 i 维的二进制 Trie 结构上进行前缀匹配,当规则与二进制 Trie 结构中某节点对应的前缀相匹配时,将该规则所在的 Tuple 对应的向量位置置 1。

4) 为每个 Tuple 哈希表设置一个标志位,例如 m 个 Tuple 哈希表,就有 m 个标志位,这 m 个标志组合成为一个 Tuple 位向量。

5) 对各 Tuple 分别通过哈希表利用 IXP2850 网络处理器的 Hash 单元处理为每个 Tuple 所包含的规则建立 Hash 索引,该 Hash 索引以位的形式存储。

例如规则集合如表 3 所示。

表 3 Tuple-BV 算法的规则集合

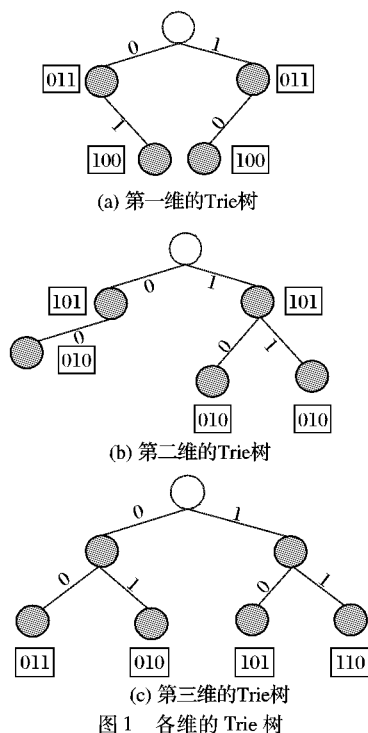
规则索引	规则描述	Tuple 规则
R1	(01 *, 1 *, 11 *)	(2, 1, 2)
R2	(1 *, 11 *, 00 *)	(1, 2, 2)
R3	(1 *, 0 *, 10 *)	(1, 1, 2)
R4	(0 *, 1 *, 00 *)	(1, 1, 2)
R5	(10 *, 0 *, 10 *)	(2, 1, 2)
R6	(1 *, 11 *, 11 *)	(1, 2, 2)
R7	(0 *, 10 *, 01 *)	(1, 2, 2)
R8	(1 *, 1 *, 00 *)	(1, 1, 2)
R9	(0 *, 00 *, 11 *)	(1, 2, 2)

通过 Tuple 空间建立索引,如表 4 所示。

表 4 Tuple 索引表

规则索引	Tuple 规则	所含规则	Tuple 位向量
T1	(2, 1, 2)	{ R1, R5 }	100
T2	(1, 2, 2)	{ R2, R6, R7, R9 }	010
T3	(1, 1, 2)	{ R3, R4, R8 }	001

各维 Trie 树如图 1 所示。



3.3 匹配过程

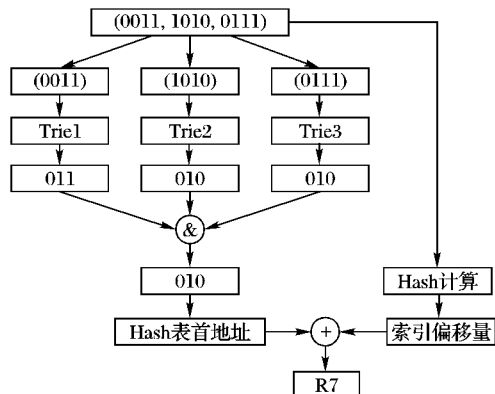
Tuple-BV 算法的匹配过程放在一个微引擎中执行,由于每个微引擎有 8 个硬件线程可以同时工作,所以可以利用硬件线程的并行特点,对匹配过程进行优化。Tuple-BV 算法的匹配过程如下。

1) 将数据包第 i 维分量在第 i 维的二进制 Trie 结构上进行搜索,第 i 维匹配可以放在当前微引擎的第 i 个硬件线程上进行,当与二进制 Trie 结构中某节点对应的前缀相匹配时,则取出该节点对应的位向量。该位向量表示数据包匹配的 Tuple 规则。

2) 将每个维度所搜索得到的位向量进行“与”操作。结果为“1”的位表示所对应的 Tuple 哈希表中有匹配的规则。如果有多个规则同时匹配,则取高位位值对应的 Tuple 规则。

查找过程可以利用网络处理器的 ffs[dest, src] 指令,按从左到右的顺序,获得源寄存器中第一个值为 1 的比特的位置。

3) 将相与的结果在 Tuple 哈希表上进行哈希操作,该哈希操作通过 IXP2850 的 Hash 单元实现,查找哈希表获取对应的规则后,根据规则的优先级,选择出最优规则。



如图 2 所示的一个数据包为 $P(0011, 1010, 0111)$, 通过对三个维度的 Trie 树进行查找依次得向量为: $[011], [010]$,

$[010]$, 相与得 $[010]$, 所以得 Tuple 空间 T_2 。再将数据包 P 按照 T_2 进行哈希运算, 找到 T_2 所在哈希表中对应的规则。

4 算法比较

假设进行一次内存访问能够读取的位数为 W , 对于 N 条 d 维的规则来说, BV 算法每一维需要进行 (N/W) 次内存访问, 在读取位向量之前, 还要通过 Trie 查找算法求得每一维上字段所在的区间, 假设这个步骤花费的时间为 t , 那么总的时间复杂度为 $O(t + N/W)$ 。 N 条规则在一维上最多产生 $2N + 1$ 个互不相交的区间, 每一区间要占用 N 位的空间, 因此, 最坏情况下, 位并行算法将占用 $d \times N \times (2N + 1)$ 的内存空间, 空间复杂度为 $O(N^2)$ 。随着规则集合的增大, BV 算法占用的空间 $O(N^2)$ 将以二次方的速度急剧增长, 同时它的查询时间 $O(t + N/W)$ 也将以线性速度增长。^[3-5]

Tuple-BV 算法通过对 N 条规则的 Tuple 空间分割成 m 个空间, BV 算法每一维需要进行 (m/W) 次内存访问, 最后的哈希匹配在网络处理器中可以只通过一次 Hash 运算获得, 因此, 总的时间复杂度可以近似为 $O(t + m/W)$ 。 Tuple-BV 算法的各维度的 Trie 树上的位向量共占用 $d \times m \times (2N + 1)$ 的内存空间。各 Tuple 的哈希表采用比特位建立存储, 因此共占用的空间约为 N , 相对于 $O(m \times N)$ 较小而被忽略。所以 Tuple-BV 算法的空间复杂度约为 $O(m \times N)$ 。

利用表 4 中的向量替换表 3 中的 Tuple 规则, 然后, 对比表 2 可知, Tuple 向量少于位向量占用的存储空间。 Tuple-BV 算法在最坏情况下, 才会有 $m = N$, 而在大型网络中, $m \ll N$, 因此的位向量远比 BV 算法短, 数据结构更紧凑, 占用的空间更少, 因此更适合网络处理器进行处理。

5 实验分析

实验采用思博伦的 TestCenter 高密度综合数据网络测试平台, 比较 BV 算法和 Tuple-BV 算法在规则数为 100、500、1000 和 2000 时对 64 B 随机数据包的处理延时, 实验结果如表 5 所示。

表5 BV 算法和 Tuple-BV 算法延时比较 μs

规则数	延时种类	BV 算法	Tuple-BV 算法
100	最小延时	9.21	9.25
	最大延时	394.38	401.85
	平均延时	213.02	218.44
500	最小延时	10.51	10.45
	最大延时	412.10	409.29
	平均延时	220.36	224.42
1000	最小延时	12.97	11.73
	最大延时	434.88	423.32
	平均延时	238.37	232.07
2000	最小延时	16.24	13.82
	最大延时	485.59	446.91
	平均延时	261.46	248.18

从表 5 实验数据可以看出, 在规则较少的时候, Tuple-BV 算法与 BV 算法的处理延时接近, 个别指标上略微低于 BV 算法, 主要原因是 Tuple-BV 算法比 BV 算法多进行了一次 Hash 处理。当规则数为 1000 和 2000 的时候, Tuple-BV 算法的位向量短, 读取次数少的优势开始体现出来, 从平均延时和最大延时上可以看出, Tuple-BV 算法明显低于 BV 算法。

(下转第 315 页)

加密及防火墙在几种常见的 VoIP 攻击手段的防范能力方面做了相关分析,结果如表 1 所示(√表示能够防范;×表示不能完全或者完全不能防范)。

表 1 安全策略的防范能力

主要攻击手段	VPN	VLAN	数据加密	防火墙
DoS	√	√	√	√
SPIT	√	×	×	√
偷听	√	√	√	√
发起无效通话	×	×	×	√
拨打免费电话	√	√	√	√
中间人攻击	√	√	×	√

可以用一些抓包工具来观察具体的一个防范措施(抓包软件采用的是 WireShark),如数据加密,这是无线终端接入无线网络时采用的技术手段。无线终端发出 EAP-Response/Identify 请求包,然后 AP 将其包发给认证器,认证服务器用指定的认证算法校验终端的合法性;如果成功,则允许其进入 AP。具体如图 5 所示。

No.	Time	Source	Destination	Protocol	Info
1	0.000000	IntelCor	BelkinIn 5f:32:02	EAPOL	Start
3	4.491700	BelkinIn	IntelCor 80:fc:2b	EAP	Request, identity [RFC3748]
5	9.490545	BelkinIn	IntelCor 80:fc:2b	EAP	Request, identity [RFC3748]
12	41.739624	BelkinIn	IntelCor 80:fc:2b	EAPOL	Key
15	41.891255	IntelCor	BelkinIn 60:c3:cc	EAPOL	Key
16	42.736544	BelkinIn	IntelCor 80:fc:2b	EAPOL	Key
17	42.739432	IntelCor	BelkinIn 60:c3:cc	EAPOL	Key
18	42.745277	BelkinIn	IntelCor 80:fc:2b	EAPOL	Key
19	42.745277	IntelCor	BelkinIn 60:c3:cc	EAPOL	Key

图 5 EAPOL 协议包

图 6 对 SIP 包进行分析,源端地址发出 VoIP 注册请求,如果它无法知道 VoIP 号码及密码,SIP 服务器拒绝此次请求,则它无法注册成功,即可达到阻止偷听、发起无效通话等其他攻击手段的目的。由于本公司的 SIP 服务器只是局限于公司 Intranet 内部,并没有进入公网,所以对其他防范由防火墙做相应的限制策略,即可防止非法用户进入公司内部网络,从而无法进行发起无效通话、DoS、SPIT 等以上列出的攻击手段。

Time	Source	Destination	Protocol	Info
5.281753	10.31.19.10	10.31.21.19	SIP	Request: REGISTER sip:10.31.21.19
5.290304	10.31.21.10	10.31.19.13	SIP	Request: OPTIONS sip:62802112610.31
5.293574	10.31.21.10	10.31.19.13	SIP	Status: 100 Trying (1 bindings)
5.293374	10.31.19.10	10.31.21.19	SIP	Status: 480 Temporarily Unavailable
5.293462	10.31.21.10	10.31.19.13	SIP	Status: 401 Unauthorized (0 bind
5.293462	10.31.21.10	10.31.19.13	SIP	Status: 401 Unauthorized (0 bind
5.402588	10.31.21.10	10.31.19.13	SIP	Request: OPTIONS sip:62802112610.31
5.402588	10.31.21.10	10.31.19.13	SIP	Status: 100 Trying (1 bindings)
5.404035	10.31.19.10	10.31.21.19	SIP	Status: 480 Temporarily Unavailable
5.475247	10.31.21.10	10.31.19.13	SIP	Status: 200 OK (0 bindings)
5.889920	10.31.19.10	10.31.21.19	SIP	Request: REGISTER sip:10.31.21.19
5.889964	10.31.21.10	10.31.19.13	SIP	Status: 100 Trying (1 bindings)
5.889794	10.31.21.10	10.31.19.13	SIP	Status: 401 Unauthorized (0 bind
6.092260	10.31.19.10	10.31.21.19	SIP	Request: REGISTER sip:10.31.21.19
6.098569	10.31.21.10	10.31.19.13	SIP	Status: 100 Trying (1 bindings)
6.099246	10.31.21.10	10.31.19.13	SIP	Status: 403 Forbidden (bad auth)

图 6 SIP 包

对于 VoIP 的安全评估目前还不太成熟,评估方法和手段都还不完善,都仅描述了 VoIP 安全的功能性,但是没有最佳

的实施方案和保护 VoIP 网络安全的指导。所以本文在此给出了对于一些基本攻击手段的防范能力分析,旨在说明本系统对于这些攻击手段的防范能力。

4 结语

随着 VoIP 技术的持续发展和逐渐成熟,VoIP 最终会成为取代传统 PSTN 技术的主要通信技术,如何确保 VoIP 的通信安全也变得越来越重要。本文可以防范基本的攻击手段,实现基本通话安全,但对其他安全攻击没有做出相应评测。通过实施完善的安全策略来抵抗软件攻击,设计相应的入侵检测系统将是今后的工作重点。

参考文献:

- [1] 司端峰,韩心慧,龙勤,等. SIP 标准中的核心技术与研究进展[J]. 软件学报,2005,16(2): 239-250.
- [2] ZHANG Y. SIP-based VoIP network and its interworking with the PSTN [J]. Electronics & Communication Engineering Journal, 2002, 14(6): 273-282.
- [3] Voice over IP Security Alliance [EB/OL]. [2009-07-06]. <http://www.voipsa.org>.
- [4] DENG D J, YEN D J. Quality-of-service provisioning system for multimedia transmission in IEEE802.11 Wireless LANs [J]. IEEE Journal on Selected Areas in Communications, 2005, 23(6): 1240-1252.
- [5] EDNEY J, ARBAUGH W A. Real802.11 Security: WiFi Protected Access and 802.11i [M]. Reading, Massachusetts: Addison-Wesley, 2003.
- [6] Signalling System #7(SS7) [EB/OL]. [2009-07-05]. <http://www.itu.int/ITU-T/>.
- [7] FRIEDMAN T, CACERES R, CLARK A. RTP control protocol extended reports, RFC3611 [S/OL]. [2009-06-25]. <http://www.rfc-editor.org/rfc/rfc3611.txt>.
- [8] WinEyeQ [EB/OL]. [2009-06-07]. <http://www.touchstone-inc.com/wineyeq.htm>.
- [9] International Telecommunication Union. The E-model, a computational model for use in transmission planning, G.107 [S], 2000.
- [10] International Telecommunication Union. Methods for subjective determination of transmission quality [S], 1996.
- [11] International Telecommunication Union. Subjective performance assessment of telephone-band and wideband digital codes [S], 1996.
- [12] International Telecommunication Union. Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs [S], 2001.

(上接第 308 页)

6 结语

本文针对 IXP2850 将 Tuple 空间分割思想结合到 BV 算法中,实现一种适用于网络处理器的改进算法——Tuple-BV 算法,该算法通过 Tuple 分割,缩短位向量长度,减少了存储读取次数,提高了包分类的处理速度,该算法适用于大型网络环境中,为网络处理器实现包分类算法提供了借鉴和经验。

参考文献:

- [1] LAKSHMAN T V, STIDIALIS D. High speed policy-based packet forwarding using efficient multi-dimensional range matching [C]// Proceedings of ACM SIGCOMM '98. New York: ACM, 1998: 203-214.
- [2] IYER S, RAO KOMPALA R, SHELAT A. Architecture for fast and flexible packet classification [J]. IEEE Network, 2001, 15(2): 33-41.

- [3] 郑凯. 高性能 IP 路由查找和分组分类技术的研究 [D]. 北京: 清华大学, 2006.
- [4] 肖小林. 基于网络处理器的包分类引擎设计与实现 [D]. 长沙: 湖南大学, 2006.
- [5] 郑裕峰. 高速包分类协处理器及网络平台研究 [D]. 合肥: 中国科学技术大学, 2007.
- [6] Intel Corporation. Intel IXP2850 Network Processor, Hardware Reference Manual [M]. [S. l.]: Intel, 2004: 1-3.
- [7] 张宏科, 苏伟, 武勇. 网络处理器原理和技术 [M]. 北京: 北京邮电大学出版社, 2004. 郑裕峰. 高速包分类协处理器及网络平台研究 [D]. 合肥: 中国科学技术大学, 2007.
- [8] SRINIVASAN V, SURI S, VARGHESE G. Packet classification using Tuple space search [C]// Proceedings of ACM SIGCOMM '99. New York: ACM, 1999: 135-46.