一种新的高维报文分类算法[1]

－－无相交树(NONIntersection Trie)算法

陆晟、龚俭
东南大学计算机科学与工程系

摘要：报文分类是网络交换设备的基础操作之一，它会在很大程度上影响相关硬件设备和软件系统的功能和性能。目前存在大量的报文分类算法，但大多数都是针对低维分类问题。随着网络管理和网络入侵检测等应用需求的日益增长，本文提出了一种新的高维报文分类算法 NI Trie，该算法具有其它报文分类算法所不具备的强表述支持能力和低时间复杂度并存的特点，其期望时间复杂度和期望空间复杂度均达到了同类分类算法的最优或接近最优。它具备较强的软硬件通用能力，可适用于较广的应用领域。实际应用性能良好。
关键字：报文分类、分类算法、网络、高速、强表达能力、无相交树。

1. 引言

网络中利用报文分类算法的场合非常多，从路由寻址、交换机设计、防火墙实现一直到入侵监测和负载均衡等。但随着网络速度的不断提高，从 FE 到 GE，从 OC48 到 OC192，报文分类算法的速度要求日益增强。特别是高速路由器等设备对于基于地址的一维分类算法提出了更高的要求。

报文的高速分类可以有软件、硬件实现的多种方法，这些方法适用于不同场合下的应用，能够以不同的代价解决不同速率网络环境下的问题。但是这些方法一般都只提供了特定维数的分类能力，特别是一维分类和二维分类算法较多；而能够提供对高维分类支持的算法要么空间复杂度太大无法满足低成本要求，要么运算复杂度无法满足高速网络环境的应用需求。最典型的一维分类应用是路由器和交换机的转发功能设计，通过查看目标 IP 地址或目标 MAC 地址决定发送端口；二维分类中最常见的是根据地址和端口进行，例如流量控制等；而高维分类在另外一些场合被使用，一般路由器提供的访问控制是根据：源地址、目的地址、协议类型、源端口、目的端口的五维分类，有的增加对方向的控制，形成六维分类，而入侵检测系统中需要作更多的分析，因此被用于分类的维也更多。

本文提出的高速分类算法适用于任意维的分类需求，可以达到 O(dlogk)的时间复杂度和 $O(nk\log_k(n))$的空间复杂度，能够提供非常灵活的表达能力，其中的 d 是维数，k 是每维相关的操作个数，而 n 是规则数目。本算法用纯软件实现，在 GE 环境中测试的运行结果良好。此外本方法也具备改为硬件实现的能力，有较好的可扩展性。本文首先介绍了目前的报文分类算法和报文分类算法的指标，然后通过一个形式化系统说明无相交树分类算法的结构和使用方法，最后本文对这个分类算法的各种指标进行了理论分析和试验验证。

本文提出的分类算法适用于防火墙、入侵检测、计费分析、流量分析等高维，规则相对稳定的环境，而对于路由转发、地址交换、MPLS 交换、动态负载均衡、流量控制等动态环境、维数较少的应用则不适用或无法达到最优。

2. 报文分类算法的技术指标

报文分类算法存在许多技术指标，例如，时间复杂度、空间复杂度和更新复杂度等，但是在这些指标中，往往通过域宽 W 加以定义，因为传统的报文分类算法一般均采用将比较

[1] 本文内容受国家自然科学基金重点课题（90104031）资助

改为前缀匹配的处理方法。但是由于本文提出的无相交树算法(NONIntersection Trie, NI Trie)，如果采用软件实现，可以不采用前缀方法，因此虽然在下一部分将仍旧采用域宽定义各种复杂度，但是 NI Trie 算法本身的度量方式却略有不同。

一般的，报文分类算法的技术指标可以定义为：

- 时间复杂度：该指标用于定义分类所需要的步骤或循环的数目的最大边界 O(f)，f 是规则个数 n、域宽 W 和维数 d 的函数。例如线性分类算法的时间复杂度是 O(n)，而 Grid-of-Trie 算法的时间复杂度是 $O(W^d)$。
- 空间复杂度：该指标用于定义分类所需要保存的规则数据库和相关数据结构所需要空间的最大边界 O(f)，和时间复杂度相同，f 也是规则个数 n、域宽 W 和维数 d 的函数。例如线性分类算法的空间复杂度是 O(n)，Grid-of-Trie 算法的空间复杂度是 O(ndW)。
- 更新复杂度：该指标用于定义对于规则数据库进行插入、删除和修改操作所需要的步骤或循环的数目的最大边界 O(f)。例如线性分类算法的更新复杂度是 O(n)，Grid-of-Trie 算法的更新复杂度是 O(nW)【1】
- 表达复杂度：该指标用于定义规则中域的数目和每个域可以支持的操作的乘积。也就是用于说明该分类算法能够承受的分类表达能力。例如一般路由器是一维前缀匹配操作，而 CISCO 路由器访问控制列表是六维前缀匹配操作。

此外，还有运算复杂度用于对运算能力的需求进行描述，运算复杂度与时间复杂度之间最大的区别在于，运算复杂度表示的是总的运算能力，因此其总是大于等于时间复杂度，一般是时间复杂度和所使用的并行处理机个数的乘积。

对于报文分类算法而言，一般期望时间复杂度、空间复杂度和更新复杂度越小越好，表达复杂度则越大越好。然而，在许多情况下，报文分类算法无法使全部指标达到最优，而是应该根据算法的使用场合加以折衷。例如对于核心骨干路由器的报文转发分类算法就需要好的更新复杂度，因为有文献指出骨干路由器的路由表可能每秒更新数百次【2】，然而对于防火墙或入侵检测系统而言，就需要更高的表达复杂度，却不一定需要高的更新复杂度，因为这些应用场合的更新速度均不会非常频繁，而且可以容忍一定程度的报文丢失。

3. 传统的报文分类算法及其应用

Pankaj Gupta 和 Nick McKeown 把报文分类算法分为四类【3】：基本数据结构算法(Basic Data Structure Algorithms (BS))、几何算法(Geometric Algorithms (GA))、启发式算法(Heuristics Algorithms (HA))、硬件算法(Hardware-Based Algorithms (HW))。作者根据目前报文分类算法的发展现状，并按算法的复杂度和应用方式对【3】的工作进行了完善和扩展，得到下表：

名称	算法说明	类型	时间复杂度	空间复杂度	应用场合说明
Table	构造全部可能情况的表，用可能情况作为随机寻址的下标	BS	1	2^w	单维分类，域宽窄
Linear	将所有规则顺序排列，一一比较	BS	n	n	规则数少
Ternary CAM	将所有规则用 TCAM 硬件实现	HW	1	n	速率要求极高，允许成本高，维数较少
Hierarchical tries	将每维分类均排列为按比特的搜索树，并将树相互连接	BS	W^d	ndW	维数一般不超过二，规则数少
Set-Pruning tries	Hierarchical tries 的改进。通过复制部分树，可以不回溯搜索	BS	dW	n^d	维数一般不超过二，规则数可较多
Grid-of tries【1】	Set-Pruning tries 的改进。将复制功能通过指针实现，节约空间	GA	W^{d-1}	ndW	二维分类，更新复杂度高

Cross-producting	通过预计算的叉乘表进行查找	GA	dW	n^d	二维分类，纯软搜索效果一般
AQT【5】	二维树查找	GA	W	nW	二维分类
FIS-tree【4】*	利用规则之间的覆盖关系构造 Fat Inverted Segment Tree	GA	(L+1)W	$Ln^{\frac{L+1}{L}}$	二维分类
RFC	分步并行运算，逐步求精	HA	d	n^d	需要并行计算能力，速度快，存储和维数相关
Bitmap-intersection	分维利用投影关系计算位表，利用位表合成运算结果	HW	dW+n/memwidth	dn^2	需要硬件实现，存储空间大
HiCuts【6】	通过几何搜索获得一个规则的子集，其中规则数目少于一常数，继续搜索获得最终规则	HA	d	n^d	空间需要量大，速度快，可多维分类，更新复杂度低
Tuple space search	预计算规则前缀元组，比较时进行 hash 匹配操作	HA	n	n	虽然最坏时间复杂度为 O(n)，但平均较快，空间省
Multibit Tries**	在一维中多比特同时搜索	BS	W/k	$\frac{2^k nW}{k}$	单维分类，主要用于地址查询
LC Tries	Multibit Tries 的层次压缩	BS	W/k	$\frac{2^k nW}{k}$	单维分类
Binary range search	用前缀表示一定范围进行搜索	BS	$\log_2(n)$	n	单维分类，主要用于地址查询
Multiway range trees ***	用二叉搜索数的范围查询	BS	$\log_2(n)$	$nk\log_k(n)$	单维分类

* FIS-tree 中的 L 表示树的层次。

** Multibit Tries 中的 k 表示同时搜索的比特数目

*** Multiway range tress 中的 k 是搜索树的节点分支数

表一 常见报文分类算法简表

在以上的各个算法中，最快的是 Table 和 TCAM 算法，均为常数时间，但是 Table 算法的空间复杂度太高，当域宽大或多域时几乎不可能使用，而 TCAM 的成本相当高，如果支持的域多，则成本会过高。而时间复杂度是规则数目 n 的多项式的算法均较慢。对于单维而言，W 必然大于$\log_2(n)$，因此除了常数分类时间的算法外，最快的算法均和 W 或$\log_2(n)$相关。因此，多维分类中的 O(dW)和单维分类中的 O(W)或 O(W/k)均为较快的算法。

对于空间复杂度而言，n 是基本值，因为无论如何需要把规则全部存储下来。因此接近于 O(n)空间复杂度的算法均较好，例如 AQT、TCAM 和 Binary range search 等。

此外，所谓 TCAM 算法仅仅指使用 CAM/Ternary CAM 所实现的硬件算法，因此不同的 TCAM 算法的搜索策略也会不同。以上有的软件算法也可以用 TCAM 实现，从而实现并行，虽然运算复杂度没有降低，但时间复杂度则会极大减小。又如 PMC-Sierra 公司的 ClassiPI 协处理芯片处理器使用 ASIC 电路，但仍旧需要有自己的内部处理算法支持。

4. NI Trie(无相交树(NONIntersection trie))的形式化系统

4.1 形式符号

Field：域 F 是报文中特定部分的比特的集合。在 NI Trie 中，域可以是任意的比特序列，但是必须相连。在分析过程中域是一种无语义概念。

Operator：操作符 O 是不和特定域相关的关系操作符或逻辑操作符。O = RelationOP |

LogicalOP。

RelationOP：关系操作符为一个集合，ROP={EQ、GT、LT、GE、LE、NE}。分别表示等于、大于、小于、大于等于、小于等于和不等于操作。

LogicalOP：逻辑操作符为一个集合，LOP={NOT、AND、OR}。分别表示非、与、或操作。

Constant：常量 C 是的整数，在实现时是小于等于 2^64-1 的整数。

Atom：原子 A 指由域、操作符、和常数所组成的集合。A={F, O, C}。

Expression：表达式 P 是由域、操作符和常量所构成的序列，也就是原子所构成的序列。P=[A1, A2, … An]。但是这个序列是有语义含义的，即采用后序表达的任意表达式，操作数的个数取决于操作符的目数。事实上，表达式所采用前序还是后序还是中序并不重要，只是通过位置表达其语义含义。

Rule：规则ρ是从表达式到布尔变量的函数映射关系，ρ: P->Boolean。当表达式的值为非 0(真)时，往往会进行一定的动作，例如，防火墙允许报文通过等。

Rule List：ℜ是规则的集合。ℜ=[ρ1, ρ2, … , ρn]。这是有序集合，因此用 ‘[]’ 代替 ‘{}’。

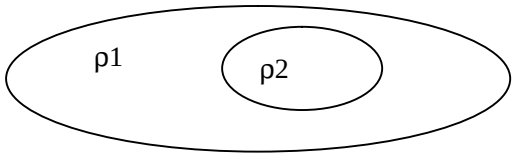
Selection Operator：选择操作是一个函数σ: ℜ->ℜ，用于选择规则集合中的一个子集。这个规则子集中的所有规则ℜ均对于特定报文为真。

Classification Function：分类函数是π从规则集、选择操作和特定的域值(实际为特定报文中和分类相关的所有特定域的值的集合)到规则的映射关系，π: (ℜ, σ, F)->ρ。该功能从特定规则集合中，在对特定报文所有为真的规则集合中，根据优先原则选择最终的匹配规则。

在以上的形式系统中使用了 Sundar Lyer 等人所采用的形式化系统中的部分名称【7】，但 Sundar Lyer 的形式化系统所支持的实际上是一个全“与”结构的规则系统，也就是规则中域(Field)之间的关系是全“与”关系。而 NI Trie 需要支持的是任意的表达式结构，因此对原来的形式化系统作了较大修改以满足该需求。

4.2 规则的覆盖和冲突

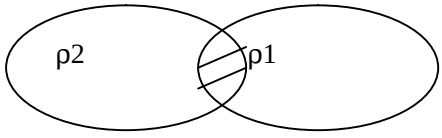
如果∃ρ1, ρ2，其中ρ1: P1->Boolean、ρ2: P2->Boolean。而 P2=[P1, Aj, Aj+1 …]，则ρ2 覆盖ρ1，记作：ρ1←ρ2 或ρ2→ρ1。如图所示：



图一 覆盖关系

ρ2 的规则长，因此所对应的可能报文集合小，由于 NI Trie 算法对于覆盖关系采用长表达式优先的方法，因此ρ2 比ρ1 优先，而无关与这两个规则的位置。从图中可知，ρ2 事实上遮住了ρ1 的部分可能结果。这个覆盖解决方案相当与前缀分类方法中的最长前缀匹配算法(longest prefix matching algorithm)。

如果∃ρ1, ρ2，其中ρ1: P1->Boolean、ρ2: P2->Boolean，如果ρ1 和ρ2 可能同时为真，则称为ρ1 和ρ2 冲突，记做ρ1↔ρ2。例如：ρ1 中 P1=[Pi, Pj]，ρ2 中 P2=[Pi, Pk]，且 Pj≠Pk，那么如果 Pj 和 Pk 有可能同时为真，则ρ1 和ρ2 可能同时为真。



图二 冲突关系

如果ρ1 和ρ2 在同一个域上出现可能同时为真的冲突，则称为强冲突，记作ρ1⇔ρ2。例

如出现的 Pj 和 Pk 分别为：Pj=[F1, C1, EQ], Pk=[F1, C2, EQ]，如果 C1≠C2，则ρ1 和ρ2 不冲突，也不会在同一个域上同时为真，这种结构我们就说在 F1 上，ρ1 和ρ2 规则是不相交的。NI Trie 算法的重要工作就是寻找在特定域上不相交的规则，而其它也需要进行该域分类的规则就是强冲突的。在下文将说明，强冲突会导致 NI Trie 分类算法效率下降，而其它的冲突存在并不会对 NI Trie 算法的效率产生影响。

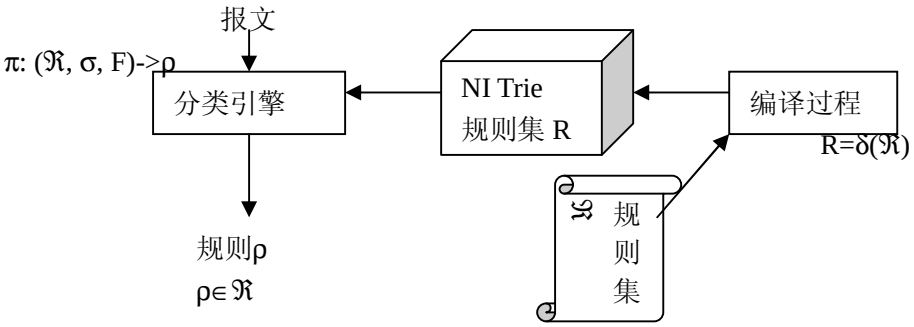
对于冲突的存在，NI Trie 采用前规则优先的原则，及如果ρ1 在规则集ℜ中排列在ρ2 前，则，ρ1 比ρ2 优先度高。在图二中的阴影属于ρ1。

5. 高速报文分类算法－－无相交树(NONIntersection trie)

和一般的全“与”结构的报文分类算法不同，本文给出的报文分类算法支持如上节所示的完整的表达式结构。而本文的算法被命名为无相交树是因为该树的结构虽然类似于 Grid-of-Trie，但并不采用按比特比较的处理方法，而是通过构造无相交集的方法加以处理。

5.1 NI Trie 的基本结构

由于 NI Trie 的强表述能力支持，因此不能够象一般的全“与”分类算法一样，简单的将规则原样进行处理，或仅仅经过简单的变换，而必须经过编译重构过程：



图三 NI Trie 的基本结构

从 NI Trie 的基本结构可知，NI Trie 实际上等价重构原规则集，也就是说，NI Trie 的规则集 R 是原规则集ℜ的等价规则集，当然此时的规则集 R 和原规则集的形式化定义并不相同。补充定义一些概念如下：

Trie Search：NI Trie 的分类操作定义为：ψ: (R, F)->C，及利用该规则集 R 和报文的特定域值得到数值 n。

Number：n∈C，是一个编号。

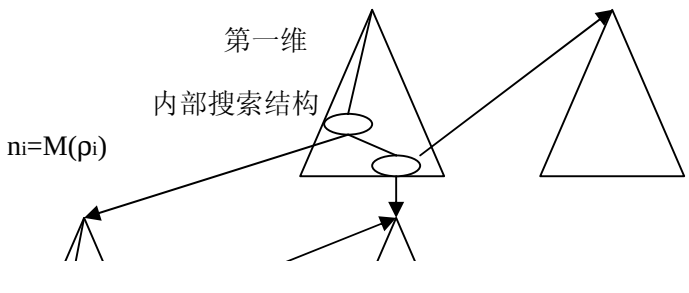
Mapping：M: ℜ->C，是从原规则集到数值的函数映射关系，但是 M^{-1} 必须存在。

因此，***“等价”(Equipollence)***关系可以定义为：对于∀n∈C，如果 n=ψ(R, F1)，而ρ=π(ℜ, σ, F2)，F1=F2，则 n=M(ρ)。

同样，可以定义编译过程(***Compiler***)为：δ: ℜ->R。如果 r=δ(γ)，则 r 和γ等价。

5.2 NI Trie 的数据结构

NI Trie 同样将比较操作用维加以表示，对于一个特定的域的比较操作就构成一维的分类。NI Trie 算法首先构造第一维的比较结构，通常是树。对于任意维 d(d>1)也同样构造比较结构，并且将这些不同维之间的比较连接成网状。需要特别注意的是每一维内部的所有比较结论是不相交的。



图四 NI Trie 的数据示意图

当在某一维中搜索到某规则匹配后，就进入相应的某维进行进一步比较。任何比较操作均有两个出口，一个为真操作，另一个为假操作，分别对应于比较为真或为假时的下一步动作。当所有比较均失败时，也存在一个出口，指向某维比较操作。

在和维比较操作等价的地位上，也可能存在结论节点，标志于特定的规则ρ。当到达结论节点 n_i 时，就说明分类操作π的结果为$\rho_i = M^{-1}(n_i)$。

以上结构是一个无回路的网状结构，和 Grid-of-Trie 的不同表现在一下三方面：

- 同一层次上的比较不一定只涉及同一个域，例如，第二层上的各个比较不一定都关于 F2，而可能和任何维相关。
- 每一维的比较不一定只出现一次，同一个域可以出现在多个不同的搜索位置上。
- 每一维中，不采用按比特比较的方式，而采用操作符和参数的表示方式，因此不能够象 Grid-of-Trie 一样从中间切分子树。

由于以上这些性质，NI Trie 可以支持任意维的操作，而不象 Grid-of-Trie 等等仅仅能够在低维分类时有较好的性质。

根据该结构可知，当规则形成 NI Trie 结构时，逻辑操作符已经被消除掉了，而真操作出口事实上表示“与”关系，假操作出口表示“或”关系，至于“非”关系，则已经表现在查找操作中了。

5.3 NI Trie 的生成算法

NI Trie 良好的分类效果和强的表达能力是建立在非常复杂的生成算法的基础上的。根据规则集ℜ形成 NI Trie 的算法，就是 NI Trie 的生成算法。

NI Trie 的生成算法由以下四步构成：

- 规则的范式化：将任意表达式修改为合取范式。
- 域的顺序重构：根据特定的方法重新构造合取范式中域的顺序关系，合理的域顺序关系可以最大限度的将任意表达式的规则加以压缩。目前有两种主要的处理方法进行域重构，一种方法是采用预定义的域优先顺序进行，例如，定义 P(F1)>P(F2)>…>P(Fn)，其中 P(x)函数是优先级定义。第二种方法是采用统计自学习的方法，在作δ操作时，临时确定域之间的顺序关系。
- 相同域的和并与最大无相交集的求解：对于相同域的搜索操作，将其中操作结果集合不相交的部分合并，形成和域 F 相关的内部搜索操作集合。
- 单域搜索结构的构造：将全部内部搜索操作集合组织成内部搜索结构。

通过以上四步以后，规则变换操作δ完成，NI Trie 生成，可以利用搜索引擎进行操作。

5.4 NI Trie 的基本搜索引擎

当报文到达时(给定特定的域值时)，NI Trie 从第一维处开始搜索，如果搜索到某操作为

真，则从该操作的“真”出口进入下一维搜索结构，否则，从所有比较均失败的出口进入下一维的搜索结构。直到达到结论节点，通过结论节点中编号 n_i 计算出$\rho_i = M^{-1}(n_i)$，从而，完成分类功能π: ($\Re$, σ, F)->ρ。

由于一般情况下 NI Trie 分类算法的存储需求远远小于基于比特的分类算法，因此，对于每维的平均分类条件数目均不大的应用场合，NI Trie 算法可以方便地改变为纯硬件实现，从而提高分类的速度。

6. NI Trie 的复杂度分析和适用范围

对于一般无冲突规则而言，NI Trie 的最坏时间复杂度为 O(dlogk)，其中 k 是最大的单维比较动作数目，k 总小于 2^w。由于 NI Trie 将比较操作直接放置在搜索过程中，因此当每一维的比较动作和 W(域宽度)相比要小时，操作动作比基于比特的搜索方法要少。如果每一维均采用树形的表示方式，在最坏情况下，全部是等于操作(NI Trie 单维中容纳最多节点的方法)，则树的深度为 $\log(2^w)=W$，和基于比特的搜索方法具有相同的数量级，但存在一个常数系数的额外开销。

但是，如果规则本身存在大量强冲突，NI Trie 的效果并不理想，任何一规则的冲突存在就会使运算时间增加一个单维分类时间 O(logk)，此时的 k 是该维的比较动作数目，当系统中有 p 比例的域冲突，冲突的规则均为 1 时，总的时间复杂度会上升到 O(d(p+logk))。

规则的冲突对于一般应用而言均存在，但比例非常低，例如一 655 条规则的入侵检测系统，其中存在冲突的规则 4 条，占总规则数的 0.61%，因此效率损失是可控的。这个例子来自 863-317 通信主题重大课题“高速 IP 网络运行监测和保障系统”所使用的规则，而这些规则从目前网络上常见的规则中翻译获得，代表了对当前主要网络入侵威胁的检测能力，因此在入侵检测领域具有典型的代表性，而在以上例子中时间复杂度仅增加 4。

对于无“或”关系的 NI Trie 的空间复杂度为 $O(nk\log_k(n))$。但是，逻辑关系中优先级特别低的“或”关系的存在会导致空间复杂度的上升，上升的速度是：每使用一次分配率作析取变换，单个规则中相关比较操作节点个数增加 1 倍或 0.5 倍(不是全部节点增加一倍，而是只有当“或”比“与”优先级低时，和该“与”“或”操作相关的节点增加 1 倍或 0.5 倍)，因此如果嵌套使用分配率，单规则的空间复杂率会以 2^d 的速率上升。但是，实际入侵检测分类操作中，“或”操作不到全部规则的 0.2%，导致的空间上升不超过 0.3%，该值同样来自对目前常见入侵检测规则的分析，通过对$\Re$规则集中的表达式进行分析，可以确定 NI Trie 的 R 规则集中的结点情况；而对于目前有的过滤式防火墙而言，规则中本身就没有“或”关系存在，例如 CISCO 等路由器产品中的访问控制列表，以及 IP chain 等软件防火墙，都是全“与”关系，并且这种关系完全吻合于【7】中的形式化表述方法。

从以上讨论中可知，NI Trie 算法实际上是利用牺牲对强冲突规则和多“或”关系表达式的处理能力来达到对更常用规则的更良好支持效果。因此，对于网络上一般的多维分类规则集而言，NI Trie 算法可以达到更加良好的效果，这种效果不是对全部规则而言，也不是对最坏情况而言的。

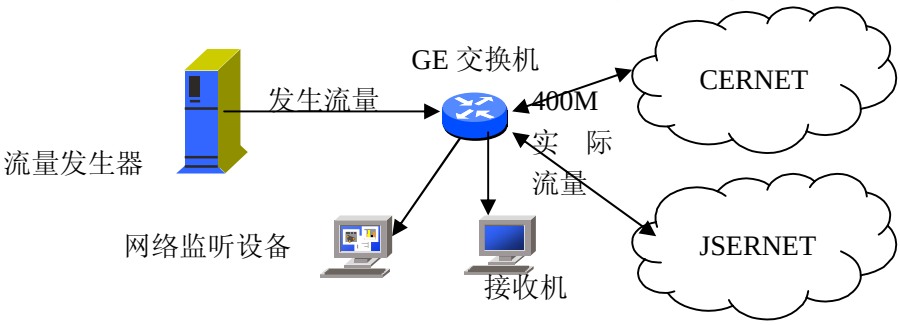
NI Trie 良好的分类效果和强的表达能力是建立在非常复杂的生成算法的基础上的。其更新复杂度是 O(nd)。

NI Trie 算法在单维分类上，比 Multibit Tries 等算法低效，在二维分类上也比不上许多

专门的二维分类算法，更新复杂度高。但 NI Trie 算法提供了灵活的多维分类能力，对网络分类应用中更加常用的几乎全“与”分类，很少的强冲突规则提供了更强的支持，适合于对更新复杂度要求不高，可是对表达复杂度要求很高的应用环境。如果每维中的操作数目均不大，例如小于 1K，则 NI Trie 算法的硬件化将使该算法在更高速的环境中应用。

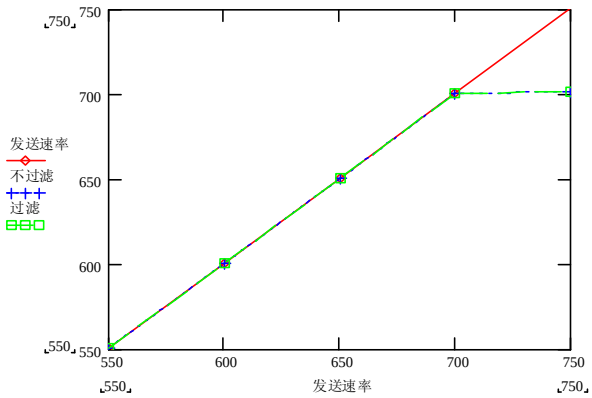
7. NI Trie 算法的软件实现测试

利用 PIII Xeon 550 * 2 的 Intel 服务器对 NI Trie 算法进行测试，使用纯软实现的 NI Trie 算法。测试的环境是 400M 正常通信流量，加 200M 到 400M 可调填充流量，其中 400M 流量是根据 GE 交换机的端口前 5 分钟平均流量获得，可调节发生流量根据流量发生器的设定与监测值相印证。全部流量的汇总通过 GE 交换机的流量镜像功能实现。试验图为：



图五 NI Trie 规则测试实验图

测试使用了入侵检测规则 655 条 (这相当于一般分类算法规则约 3K。也就是规则中所涉及的域的数目和规则数目的乘积)，当使用和不使用 NI Trie 算法时，所捕捉到的流量分别在图中用“+”和“ ”表述：



图六 使用 NI Trie 的效果曲线

从测试结果可以发现，NI Trie 算法在 PIII Xeon 550 Intel 服务器的 IO 能力下，几乎不会引起任何的性能下降，因此 NI Trie 的实际处理能力应该在以上测试速度以上，已经使这种高速服务器的 IO 成为了报文分类处理的瓶颈。当然，这种处理能力和 CPU 的处理能力相关，随着 CPU 速度的提高，NI Trie 算法将能够提供对更高的分类速率的支持。

8. 结论

本文介绍了一种新的高速报文分类算法－无相交树算法(NI Trie)，该算法能够软件实现，也能够硬件实现，其中软件实现可以提供更快的实现速度和对更复杂情况的支持，硬件实现能够提供更高速的支持，但是对维的数目和每维可以支持的分类操作数有一定限制。

由于 NI Trie 中维平均动作数 k 低，在以上的实际规则例子中为 1.545，最大 k 值为 45，所以即使以最坏的存储复杂度计算，$nk\log_k(n)$ 也远远小于一般的前缀分类算法。如果将 k

用2^w的最坏可能性计算，无冲突情况下的时间复杂度也不过 O(dW)，可以达到大多数专用二维分类算法的水平。所以 NI Trie 算法具有期望存储复杂度低，期望时间复杂度低的优点。

但是该算法作为高速分类算法的一种，不能够提供对所有应用场合的良好支持，例如，时间复杂度无法达到 TCAM 等专门硬件算法的程度。然而对相对静态的规则集合，特别使对表达复杂度要求非常高的场合，能够提供良好的支持；实际成本低，如果采用纯软件实现，则不需要额外成本；如果使用纯硬件实现，由于不再依赖全并行处理，因此成本将低于 TCAM，可以支持的维数多，并可以支持不确定维分类，虽然每维的操作数目会比全并行的 TCAM 算法低。试验证明软件实现的该分类算法至少在千兆环境环境中能够超过 66M 外频 64 位 PCI 总线的 IO 速率。

在实际使用中满足以上条件的一种重要应用就是入侵检测功能，NI Trie 提供的强表达能力，特别适合于识别攻击特征，尤其是特殊位置的特征，例如在应用层进行分析。目前在“高速 IP 网络运行监测和保障系统”中，使用了 600 多条规则，在平均 450M 流量下每天能够检测到数万条安全事件。虽然仍然存在误报问题，但这是规则表达和决策问题，不是分类问题。此外，NI Trie 对防火墙等设计也能够提供强大支持。由于在实际的多维应用场合下，k 值均很小，就导致了 log(k)远小于 W，因此 NI Trie 算法的平均运算速度比目前较好的 O(dW)要快许多(LC Tries 等能够达到 O(W)、O(W/k)和 O(log(n))仅仅因为其处理的是单维分类或确定的二维分类)。由于 k 很小，$O(nk\log_k(n))$也属于较佳的空间复杂度，和单维分类算法中的 Multiway range trees 相同(k 的含义也类似)。此外，简单的计费分析和流量分析仅仅查看 IP 地址和端口，然而，随着应用的发展，有时需要提供不同类型服务的计费，应用级流量分析也逐渐增加，此时 NI Trie 算法就显示出其强表达能力的优势。

该算法实际上是通过牺牲一些对不常用情况的性能支持，来提高分类的平均速率和减少平均存储空间，因此它是一个具有强表达复杂度的实用分类算法。

参考文献

【1】 V.Srinivasan et al., “Fast and Scalable Layer four Switching,” Proc. ACM Sigcomm, Sept. 1998, pp. 203-14.

【2】C.Labovitz, “Scalability of the Internet Backbone Routing Infrastructure,” Ph.D. thesis, Univ. of Michigan, 1999

【3】 Pankaj Gupta, NIck McKeown, “Algorithms for Packet Classification”, IEEE Network, March/April 2001, pp.24.

【4】 A.Feldman and S. Muthukrishnan, “Tradeoffs for Packet Classification”, Proc. INFOCOM, VOL.3, Mar. 2000, pp.1193-1202

【5】 M.M. Buddhikot, S. Suri, and M. Waldvogel, “Space Decomposition Techniques for Fast Layer-4 Switching”, Proc. Conf. Protocols for High Speed Networks, Aug. 1999, pp. 25-41

【6】 P. Gupta and N. McKeown, “Classification Using Hierarchical Intelligent Cuttings”, Proc. Hot Interconnets VII, Aug. 1999, Stanford, also available in IEEE Micro, VOL.20, NO.1, Jan./Feb. 2000, pp.34-41

【7】 Sundar Lyer, Ramana Rao Kompella, Ajit Shelat, “ClassiPI: An Architecture for Fast and Flexible Packet Classification”, IEEE Network, March/April 2001, pp.33-41

A Multi-Dimension Packet Classification Algorithm

------(NONIntersection Trie)

LU Sheng GONG Jian
Department of Computer Science and Engineering, Southeast University

Abstract: Packet classification is a fundamental operation performed in networking equipment such as switchers and routers. The ability of packet classification algorithm would definitely effect the function and performance of such hardware equipment. There are many packet classification algorithms existed, but most of them are focusing on low dimension classification. With the booming of the requirement of network manage and network intrusion detection, a new high dimension packet classification algorithm is given in the paper, named as NI Trie. The algorithm has an important feature that it owns both high rule complexity and low time complexity, which other packet classification algorithms do not possess. Compared with the same type of classification algorithms, the expected time complexity and expected storage complexity of NI Trie reach the optimal level or nearly reach the optimal level. It could be implemented in software or hardware, and be used in a wide range.

Keywords: Packet Classification, Classification Algorithm, Network, High Speed, Wide Bandwidth, High Rule Complexity, NI Trie