国 防 科 技 大 学 学 报
第 22 卷 第 6 期 JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY Vol.22 No.6 2000

文章编号：1001-2486（2000）06-0052-05

一种用于区分服务路由器的 crossbar 调度算法*

孙志刚，卢锡城

（国防科技大学计算机学院，湖南 长沙 410073）

摘 要：宽带路由器一般采用 crossbar 交换开关。Crossbar 交换开关对 QoS（Quality of Service）的支持十分重要。提出一种支持 IETF 区分服务（Differentiated Service）服务模型的 crossbar 调度算法——OSP（Output Serial Polling），该算法带宽利用率高，支持报文优先级。与目前存在的同类算法相比，具有硬件实现简单的优点。

关键词：路由器；区分服务；crossbar；调度

中图分类号：TN393 **文献标识码**：A

A Crossbar Scheduling Algorithm for DiffServ Routers

SUN Zhi-gang，LU Xi-cheng

（College of Computer，National Univ. of Defense Technology，Changsha 410073，China）

Abstract：High performance routers often use crossbar switch. It is important to a crossbar switch to support QoS（Quality of Service）. This paper presents a new crossbar scheduling algorithm-OSP（Output Serial Polling）which supports the DiffServ（Differentiated Service）model purposed by IETF. OSP algorithm can deal with cells that have different priority while achieving high crossbar throughput at the same time. Comparing other algorithms，OSP can be implemented more easily in hardware.

Key words：router；DiffServ；crossbar；scheduling

Internet 的迅速发展对路由器提出了新的要求。路由器一方面要支持越来越高的链路速率；另一方面还要提供一定的服务质量保证。采用 crossbar 交换开关可提高路由器的交换带宽，然而 crossbar 开关要求报文在输入端口缓冲，而 IETF 提出的区分服务和集成服务（Integrate Service）QoS 模型则要求报文在路由器的输出端口调度。因此输入缓冲的 crossbar 交换开关如何支持 QoS 成为目前路由器研究中的一个热点。

从目前研究的情况看，crossbar 开关对 QoS 的支持有三种方法。一是提高开关内部加速比，当 crossbar 的内部交换速率为链路速率的两倍时，使用相应的调度算法可以模拟输出缓冲交换开关[1]，从而支持报文在输出端口调度。第二是采用带宽预约的 crossbar 调度算法，如 WPIM[2] 等。第三种方法是采用支持信元优先级的 crossbar 调度算法，如 ESLIP[3]。高优先级的报文优先通过交换开关，从而实现区分服务。算法复杂，硬件实现困难的缺点影响了上述各种方法在宽带路由器中使用。为了解决这一问题，本文提出一种支持报文优先级的调度算法——OSP，该算法具有与 ESLIP 几乎相同的性能，但硬件实现比 ESLIP 简单。

本文首先介绍区分服务路由器和 crossbar 交换开关模型，然后提出 OSP 算法，并分析了算法的性能和实现复杂性。

1 区分服务路由器模型

区分服务是 IETF 提出的一种 QoS 模型，它主要利用 IP 报文头中的 DS（Differentiated Service）域来指定该报文对带宽、延时和丢失率的要求。路由器对 DS 域不同的报文采用不同的处理方式（PHB—Per－Hop Behavior）。为了得到区分服务，用户必须事先与其 ISP（Internet Service Provider）建立服务等

* 收稿日期：2000-05-31
基金项目：国家 863 基金项目（863-300-01-03-99）
作者简介：孙志刚（1973-），男，博士生。

级约定，以指明希望接受的区分服务类型和数据流量，并在发送报文时设置报文的 DS 域（DS 域也可在与用户相连的第一个路由器上设置）。ISP 则在网络入口处根据用户的服务等级约定对其报文进行分类（classified）、警管（policed）和整形（shaped）。当报文从一个 ISP 域进入另一个 ISP 域时，其 DS 值可能会被两个域间建立的服务等级约定修改。通过设置不同的 DS 值，用户可得到低延时、低抖动的 Premium 服务[4]、比 Best Effort 可靠性更高的 Assured 服务等[4]。由于区分服务路由器只需保存每种服务的状态信息，因此可扩展性好，适合用在骨干网络中。

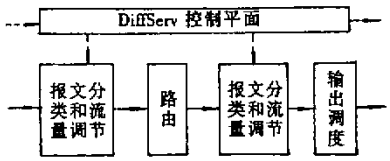


图 1 区分服务路由器模型

Fig.1 Model of DiffServ Routers

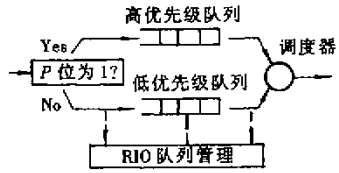


图 2 输出端口报文的缓冲和调度

Fig.2 Buffering and scheduling in output port

除了正常的路由部件外，区分服务路由器还包括区分服务控制平面（DCP—DiffServ Control Plane）、端口的报文分类和流量调节部件（TCB—Traffic Conditioning Block）以及输出端口的报文缓冲和调度部件，如图 1 所示。DCP 负责区分服务的配置和监控，报文分类和 TCB 的功能是根据报文的 DS 域对其进行分类、并对每类报文流进行警管和整形，输出端口报文缓冲和调度器对采用不同 PHB 的报文采用不同的排队、丢弃和调度策略。例如，一个简单的支持 Premium、Assured 和 Best Effort 服务的报文缓冲和调度器如图 2 所示。其中要求低延时、低抖动的 Premium 报文在高优先级队列中缓存，Assured 报文和 Best Effort 报文在低优先级队列中缓存。低优先级队列采用 RIO 算法[4]确定何时丢弃 Assured 报文，何时丢弃 Best Effort 报文。调度器采用优先级调度算法，只要高优先级队列非空，就不会调度低优先级报文。

2 支持优先级的 crossbar 交换开关模型

支持优先级的 crossbar 交换开关模型如图 3 所示。其中 $P \geqslant 1$，1 为最高优先级，P 为最低优先级。开关包含 N 个输入端口和 N 个输出端口。变长的报文进入交换开关前首先被分割成定长的信元。信元的优先级与原报文的优先级一致。开关交换一个信元的时间称为一个时间槽，第 n 个时间槽开始的时间称为时刻 n，$n=1$，2，…。若输入端口 i 到达一个目的端口为 j，优先级为 k 的信元（$1 \leqslant i, j \leqslant N$，$1 \leqslant k \leqslant p$），，那么该信元将被放入该端口虚拟输出队列 VOQ_j 的第 k 个子队列 $Q_{i,j,k}$ 中。记队列 t 时刻队列 $Q_{i,j,k}$ 的长度为队列 $L_{i,j,k}$（t）。

调度器执行调度算法，确定每个时间槽内输入输出端口间的连接关系。在第 n 个时间槽开关对信元交换的同时，调度器根据各队列的状态确定第 $n+1$ 个时间槽开关的拓扑，并将配置信息广播到 crossbar 交换阵列和输入端口控制器中。第 n 个时间槽结束后，交换阵列根据预先的配置迅速改变内部拓扑，并启动第 $n+1$ 个时间槽的交换。

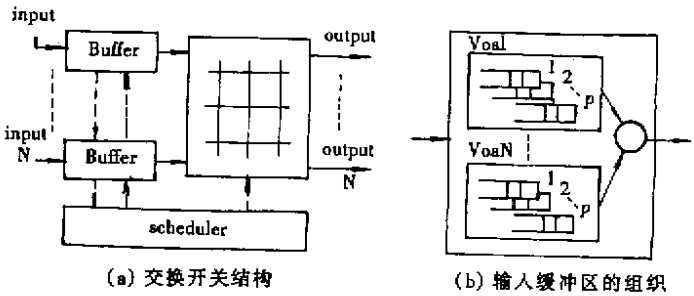
定义 crossbar 准优先级调度算法

同时满足下面两个条件的调度算法称为 crossbar 优先级调度算法：

（1）对于任意时刻 t，若调度 $Q_{a,b,c}$ 中的信元，则必有 $L_{a,b,x}$（t）$=0$，$1 \leqslant x \leqslant c-1$。

（2）对于任意时刻 t，若调度队列 $Q_{a,b,c}$ 和 $Q_{d,e,f}$ 中的信元，$a \neq d$，$b \neq e$，则必有 $L_{a,e,x}$（t）$=0$，$1 \leqslant x \leqslant f-1$ 或 $L_{d,b,y}$（t）$=0$，$1 \leqslant y \leqslant c-1$。

crossbar 准优先级调度算法不是优先级调度算法。例如，对于 $N=P=2$ 的开关，设 t 时刻有：



(a) 交换开关结构 (b) 输入缓冲区的组织

图3 支持优先级的 crossbar 交换开关模型

Fig.3 crossbar switch model with priority surpported

$L_{1,1,1}(t)>0$，$L_{1,1,2}(t)=0$，$L_{1,2,1}(t)>0$，$L_{1,2,2}(t)=0$，$L_{2,1,1}(t)=0$，$L_{2,1,2}(t)>0$，$L_{2,2,1}(t)=0$，$L_{2,2,2}(t)>0$，那么使用优先级调度算法和准优先级调度算法的方案见图4。图4表明，优先级调度算法只能利用50%的带宽，而准优先级调度算法可100%利用 crossbar 带宽。所以使用准优先级调度算法可提高交换开关的利用率。因此，对使用 crossbar 开关的区分服务路由器来说，应采用准优先级调度算法。衡量准优先级调度算法的标准是 crossbar 带宽利用率、高优先级信元在缓冲区中等待的平均延时以及算法硬件实现的复杂性。

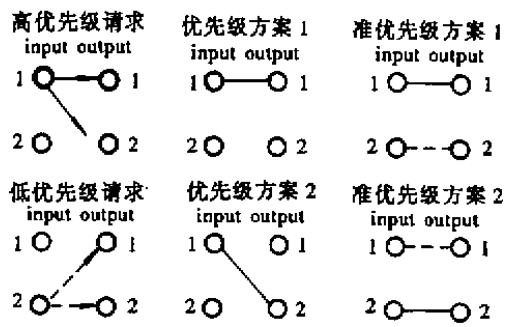


图4 调度例子

Fig.4 An example of crossbar scheduling

3 OSP 调度算法

OSP 算法采用串行轮询的思想[5]，为输出端口 i 的每个优先级设置1个 Round Robin 指针 $R_{i,k}$，指向该输入端口优先级为 k 的 N 个队列中，应优先选择调度的队列，$1\leqslant R_{i,k}\leqslant N$，$1\leqslant i\leqslant N$，$1\leqslant k\leqslant P$。OSP 算法的定义如下：

输入：

时刻 t（第 t 次调度）和第 i 个端口的请求矩阵 $REQ_i^T=[req_1, req_2\cdots req_p]$，$i=1,2\cdots N$，其中 $req_k=[req_{k,1}, req_{k,2}\cdots req_{k,N}]$，$1\leqslant k\leqslant P$。若 $L_{i,j,k}(t)>0$，则 $req_{k,j}=1$，否则 $req_{k,j}=0$。

输出：

输入输出端口的匹配集合 $\Omega=\{<i, i>, 1<i, j<N\}$，$<i, j>\in\Omega$ 表示将输出端口 j 分配给输入端口 i。

算法开始：

第一步： if ($t==1$)

for ($i=1$; $i<N+1$; $i++$)

for ($k=1$, $k<P+1$; $k++$)

$R_{i,k}=1$;

第二步： $s=t\bmod N+1$; $\Omega=\phi$; $\Psi=\{1, 2\cdots N\}$; $REQ=REQ_s$;

第三步： for ($i=0$, $i<N$; $i++$) {

out = $(s+i)\bmod N+1$; in = 0; pri = 1;

while (in == 0 && pri $\leqslant P$) {

```
            in = MA（req_pri，R_in,pri，Ψ）；pri + +
        }
        if（in≠0）{
            Ω = Ω∪ {〈in，out〉}；Ψ = Ψ - in；
            if（i = = 0）R_in = （in + 1）mod N + 1；
        }
    }
```

第四步：输出 Ω

算法结束

其中函数 MA（req_{pri}，$R_{in,pri}$，Ψ）**根据端口** out **优先级为** pri **的请求** req_{pri}，Round Robin **指针和空闲输入端口集合** Ψ **为端口** out **选择一个输出端口。若返回值** in **为** 0，**则说明匹配失败，否则** in **为匹配的输出端口号。根据算法中的** While **语句可知，对于任意时刻** t，**若算法调度** $Q_{a,b,c}$ **中的信元，则必有** $L_{a,b,x(t)} = 0$，$1 \leqslant x \leqslant c-1$。**又假设** t **时刻算法调度队列** $Q_{a,b,c}$ **和** $Q_{d,e,f}$ **中的信元，且调度过程中输出端口** b **比** e **先轮询，由于输出端口** b **选择队列** $Q_{a,b,c}$，**根据算法中的** while **语句，必有** $L_{d,b,y}$（t）$= 0$，$1 \leqslant y \leqslant c-1$，。**因此** OSP **是一种准优先级调度算法。**

4 算法模拟结果

我们对 OSP **和** ESLIP **算法的性能进行了模拟和比较。有关模拟环境的说明如下：**

（1）信元到达服从 ON/OFF burst **模型，** burst **长度服从参数为** 10 **的几何分布。** burst **信元到达每个输出端口的概率相同。**

（2）模拟的时间区间为 [1，100000]，**区间** [1，50000] **为‘预热’时间，取区间** [50000，100000] **内通过** crossbar **的信元计算统计信息。**

（3） crossbar **端口数** $N = 8$，**算法支持优先级数** $P = 2$。**带宽利用率用平均每次调度匹配的端口数表示，算法的延时用信元通过开关的平均等待时间（单位为时间槽）表示。**

（4）由于端口负载较低时，各种调度算法都相当于一种随机调度算法，在性能上几乎没有差别，因此实验仅比较端口负载在 0.7 **到** 1.0 **之间的情况。**

4.1 各端口负载一致，高低优先级信元比例相同的情况

当端口高低优先级信元比例为 1∶1 **时，算法的带宽利用率和延时特性如表** 1 **和图** 5 **所示。从表** 1 **可看出：两种算法的带宽利用率几乎相同，负载达到** 1.0 **时，** OSP **算法的端口利用率略高。**

表 1 **算法的带宽利用律比较**

Tab.1 Comparison of utility values of two algorithms

负载情况			算法平均每次调度匹配端口数	
总负载	高优先级	低优先级	OSP	PRI_SLIP
1.00	0.50	0.50	7.982	7.952
0.95	0.47	0.48	7.599	7.599
0.90	0.45	0.45	7.202	7.202
0.85	0.42	0.43	6.801	6.801
0.80	0.40	0.40	6.398	6.398
0.75	0.37	0.38	5.994	5.994
0.70	0.35	0.35	5.594	5.594

由图 5 **所示的延时特性看，** OSP **算法调度的高优先级信元延时较大（差别不大于** 2 **个时间槽），** ESLIP **算法调度的低优先级信元延时较大，尤其在负载较高时差距明显。例如当负载为** 0.95 **时，** OSP

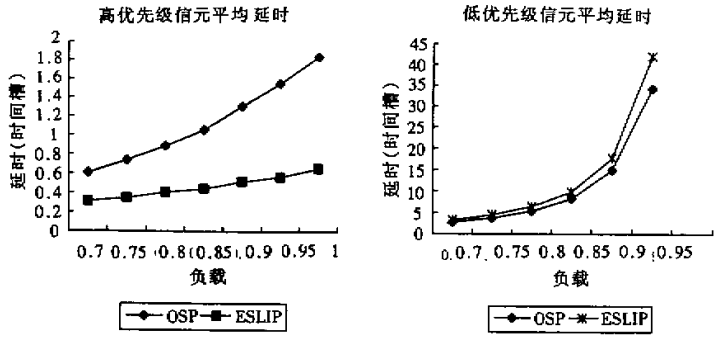


图5 算法的延时特性比较

Fig.5 Comparison of delay properties of two algorithms

和 ESLIP 算法调度的低优先级信元延时分别为 34.17 和 42.04，当负载为 1.0 时，两者分别为 608.1 和 1195。

4.2 其它情况

我们还对两种算法在其它情况下（端口负载不同、高低优先级信元比例不同）的性能进行了模拟，从模拟的结果看，两种算法对带宽的利用率几乎相同，但在一些情况下 OSP 算法调度高优先级信元的延时较短（相应地，低优先级信元延时较长），在另外一些情况下，OSP 算法调度低优先级信元的延时较短（相应地，高优先级信元延时较长）。在进行了大量的模拟和比较后，我们认为 OSP 算法的性能与 ESLIP 相当。

5 OSP 算法的实现

由于同样是串行轮询算法，OSP 调度器的结构与 ISP 调度器的结构几乎相同，因此 OSP 算法的实现可参见文献 [5]。两者在实现上的不同在于：由于 OSP 算法支持信元优先级，故需要 P 个微仲裁器同时对 P 个不同优先级上的请求进行仲裁，最后用 P 选 1 开关选择仲裁结果。

与 ESLIP 调度器比较，OSP 调度器在实现上具有如下优点：

(1) 逻辑面积小，仅需 P 个微仲裁器，而 ESLIP 需要 $2N$ 个。

(2) 更方便用流水方式实现端口调度和 crossbar 配置过程，简化配置接口的设计。

6 结束语

本文提出的 OSP 调度算法具有性能与 ESLIP 算法相当，但硬件实现简单的优点。因此可用于宽带区分服务路由器交换开关的设计中。

参考文献：

[1] Balaji Prabhakar, Nick McKeown. On the Speedup Required for Combined Input and Output Queued Switching [R]. Technical Report CSL-TR-97-738, Stanford University, 1997.

[2] Dimitrios Stiliadis, Anujan Varma. Providing Bandwidth Guarantees in an Input – Buffered Crossbar Switch [C]. IEEE INFOCOM '95.

[3] Nick Mckeown. Fast Switched Backplane for a Gigabit Switched Router. [EB]. http://www.cisco.com/warp/public/cc/cisco/mkt/core/12000/tech/fasts_wp.pdf.

[4] K. Nichols, V. Jacobson, and L. Zhang. A Two – bit Differentiated Services Architecture for the Internet [EB]. ftp://ftp.ee.lbl.gov/papers/dsarch.pdf.

[5] 孙志刚，苏金树，卢锡城. 高效的 crossbar 调度算法—ISP [J]. 计算机学报，2000，23 (10)：1078 ~ 1082.