可行的基于 CIOQ 的并行分组交换结构

任涛，兰巨龙，扈红超
（国家数字交换系统工程技术研究中心，河南 郑州 450002）

摘 要：针对传统并行分组交换结构存在的平面可扩展性问题，提出一种可行的分布式并行分组交换 PDPPS (practical distributed parallel packet switch)。在端口数为 *N* 和中间层平面数为 *K* 的情况下，PDPPS 的复用器中只需要维护大小为 *NK* 的高速缓存，就能保证每条流按序输出。理论分析和仿真结果表明，PDPPS 的性能优于使用 OQ(output queuing)结构作为中间层平面的分布式并行分组交换结构 VIQ PPS(virtual input queuing parallel packet switch)，略微低于集中式 PPS 和 IOQ PPS(in-order queuing parallel packet switch)。但相对于集中式 PPS，PDPPS 使用了更为通用且易于实现的 CIOQ(combined input and output queuing)作为中间层平面；相对于 IOQ PPS，PDPPS 使用了分布式调度算法，从而消除了系统的通信开销，并且 PDPPS 极大地降低了所需的高速缓存数量。

关键词：分布式；并行分组交换；联合输入输出排队；调度算法；保序
中图分类号：TP393 **文献标识码**：A **文章编号**：1000-436X(2011)05-0014-08

Practical parallel packet switch architecture based on CIOQ

REN Tao, LAN Ju-long, HU Hong-chao
(National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 450002, China)

Abstract: For the plane extension problem of conventional parallel packet switch, a practical distributed parallel packet switch (PDPPS) was proposed. For a PDPPS with *N* ports and *K* central planes, each multiplexor only needed to maintain *NK* high-speed buffers to guarantee the orderly transmission of each flow. Theoretic analysis and simulation results show that the performance of PDPPS is better than that of VIQ PPS(virtual input queuing parallel packet switch) whose central plane is OQ(output queuing), and slightly worse than that of centralized PPS and IOQ PPS(in-order queuing parallel packet switch). But compared with centralized PPS, the PDPPS makes use of CIOQ(combined input and output queuing) as central plane which is more common and easier to implement, compared with IOQ PPS, the PDPPS schedules in a distributed way, which eliminates communication overhead. And the PDPPS dramatically reduces the amount of required high-speed buffer.

Key words: distributed; PPS; CIOQ; scheduling algorithm; keep sequence

1 引言

近些年来，随着 WDM (wavelength division multiplexing)被广泛地应用于传输领域，链路速率已经由 10Gbit/s(OC192)增长到 40Gbit/s(OC768)，甚至达到 160Gbit/s(OC3072)。然而 DRAM 速度的发展相比较而言却很缓慢；目前主流 DRAM 的随机访问时间为 50ns 左右。巨大的速度差导致无法使用 DRAM 在一个时隙（链路传输一个信元所需要的时间）内完成一个信元的写入或读出操作。受限

收稿日期：2010-05-07；**修回日期**：2011-03-21
基金项目：国家重点基础研究发展计划（“973”计划）基金资助项目（2007CB307102）
Foundation Item: The National Basic Research Program of China (973 Program) (2007CB307102)

于 N 倍加速问题，传统的输出排队或共享缓存的交换结构已经无法满足下一代交换机和路由器对带宽日益增长的需求。由于输入排队交换结构的内部存储器和交换结构只需要工作于线速率，大部分的高性能路由器(研究[1]和商用[2,3])都选择了输入排队交换结构。输入排队交换的实质是在输入端口和输出端口之间寻找匹配，因此用于计算匹配的调度器成为输入排队交换结构的核心部分。典型的极大匹配算法有 iSLIP[4]、FIRM[5]、DRRM[6]，复杂度为 $O(N\log N)$，但至少需要 2 倍的加速才能在非均匀流量模型下达到 100%的吞吐率。最大权值匹配在任何可允许的（admissible）流量模型下不需要加速就能获得 100%的吞吐率[7]，但其复杂度却达到 $O(N^3)$。随着端口数的增加以及线速率的增加，这样的调度算法变得几乎无法在一个时隙内完成。尽管通过采用基于帧的调度[8]、流水化调度[9]等方法能够允许每次调度不需要在一个时隙内完成，但是由于单个芯片所能封装引脚数目有限，以现有的芯片制造工艺和水平将单级输入排队交换结构扩展到 T 比特以上交换容量是比较困难的。

因此，在许多可行的交换设计中，T 比特交换结构是由多个低容量交换设备组成。作为一种比较流行的设计方案，并行分组交换（PPS）在近几年来受到广泛的关注和研究[10,11]，它能够解决线速率和存储器速率发展差异问题，并提供高速率和大容量交换支持。在 PPS 的设计过程中，有 2 个关键的问题需要解决：1）PPS 调度器所处的位置，即是使用一个全局统一的调度器还是使用多个独立运行的分布式调度器。在调度器的选择和设计过程中一方面需要考虑所使用的调度器能否为每条流提供保序和负载均衡等功能，另一方面也需要考虑所使用的调度器的硬件实现复杂度和算法时间复杂度等问题。2）中间层平面的选取。一个合适的中间层平面应当尽可能提供类似于 OQ 结构的无阻塞交换，同时还需要具有较好的端口数目可扩展性和速率可扩展性。同时，如果该平面结构已经在商业领域获得广泛应用，则可以充分利用已有资源，降低整个 PPS 的成本。

2　相关工作

借鉴 SCIMA[12]的思想，Iyer 提出了 PPS 交换结构。由于最初的想法是为了将多个低速小规模的交换设备扩展成为高速大容量的交换系统，因此采用了相对简单的集中式 PPS[13]。这种 PPS 包含了一个集中式的调度器；这个调度器的时间复杂度以及与 PPS 其他部件的通信量会随着端口数和平面数的增加而急剧增加，因此集中式 PPS 的扩展性不好，并且无法在各交换平面之间均衡负载，存储资源使用低效。

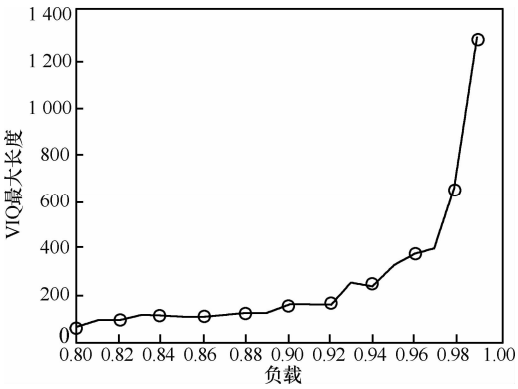
为了解决集中式 PPS 中存在的以上问题，Iyer 提出了分布式 PPS[14]的思想。分布式 PPS 是在解复用器和复用器中引入了固定尺寸的缓存，并使每个解复用器独立地执行信元分派算法以及每个复用器独立地执行信元重组算法。对于解复用器中由于内部链路被占用而无法被直接发送到相应交换平面的信元，则被暂时存储在解复用器中的相应队列中；对于复用器中由于无法满足流顺序而输出的信元，则被暂时存储在复用器中的相应队列中。但为了控制复用器中每条流的信元乱序程度，Iyer 在中间层平面的入口处引入了“最大时延”，导致整个分布式 PPS 的平均时延被大大增加。此外，分布式 PPS 的复用器中存在“死锁”问题。

VIQ PPS[15]的提出，解决了分布式 PPS 中存在的死锁问题，并保证了每条流的信元按序发送。它是让解复用器将每条流的信元按序轮询发送到各个平面，让复用器按序轮询从各个平面读取每条流的信元；同时在复用器中引入虚拟输入队列 VIQ，用于存储那些已经被内部链路发送到复用器但还未获得按序轮询输出机会的信元。然而 VIQ PPS 所使用的中间层平面依然是和集中式 PPS、分布式 PPS 相同的 OQ 结构。OQ 结构存在严重的 N 倍加速问题；即使中间层平面的运行速率是外部链路速率的 $1/K$，随着端口数 N 的增加，中间层平面依旧会变得无法正常运行。

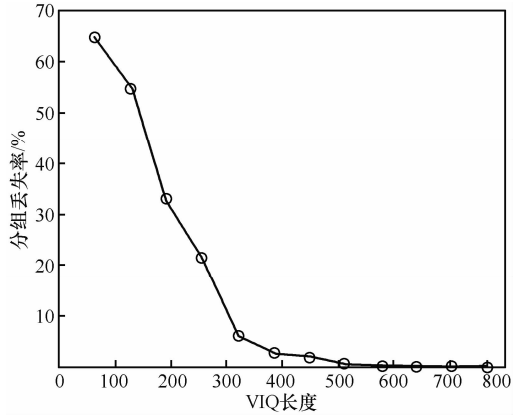
文献[16]中的 PSA 和文献[17]中的 IOQ PPS 都使用了 CIOQ 作为中间层交换平面。输入端带 VOQ 的 CIOQ 在 2 倍加速的情况下就能够模仿一个 FCFS-OQ 结构[18]，并且商业领域 CIOQ 交换结构已经被大量生产并获得广泛应用，因此使用 CIOQ 作为中间层平面是一个不错的选择。但 PSA 和 IOQ PPS 的问题是：使用了一个集中式调度器，增加了算法时间复杂度；且由于调度器和解复用器、中间层平面以及复用器之间都需要进行通信，增加了硬件实现的复杂度。

无论是集中式 PPS、分布式 PPS 还是 VIQ PPS，中间层交换平面使用的都是需要 N 倍加速的 OQ 结

构；PSA 和 IOQ PPS 虽然使用了易于实现的 CIOQ 结构，但却使用了集中式调度以满足流的保序性需要。为了考察是否能够在 PPS 中以 CIOQ 结构作为中间层交换平面的同时使用分布式调度算法，直接以 VIQ PPS 为原型，将其中的 OQ 结构改为 CIOQ 结构作为中间层平面，进行了仿真实验。实验环境是 N=64，K=10，S=2，CIOQ 结构使用了 2 倍加速的 iSLIP 匹配算法，输入流量模型为突发非均匀模型，突发长度为 32，目的端口的非均匀分布服从 $\lambda_{i,i}$=0.67ρ、$\lambda_{i,i+1}$=0.33ρ（ρ 为业务负载），仿真时间为 1 000 000 时隙。图 1(a)给出的是在不同的负载下，复用器中的 VIQ 队列长度的最大值情况。当负载达到 0.98 时，VIQ 队列的长度至少为 650 才能保证不发生分组丢失。以单个信元的大小为 64byte 计算，复用器所需要的高速缓存的大小为 650×64×10×64×8=203Mbyte。图 1(b)给出了负载为 0.98 时不同 VIQ 长度情况下的分组丢失率。如果要使分组丢失率低于 1%，VIQ 队列的长度至少需要达到 500。以单个信元的大小为 64byte 计算，复用器所需要的高速缓存的大小为 500×64×10×64×8=156Mbyte。



(a) 不同负载下 VIQ 的最大长度



(b) 负载 0.98 时不同 VIQ 长度下的分组丢失率

图 1 使用 CIOQ 作为交换平面的 VIQ PPS

当交换端口数增加、负载更大或者输入流量模型的目的端口非均匀分布程度更高时，所需要的 VIQ 队列长度还会更大。这就意味着可能无法在单个芯片上集成如此大量的高速缓存。出现这种需要大量高速缓存的原因是，在中间层平面为 CIOQ 结构且使用 2 倍的内部链路加速的情况下（不使用内部加速时整个 PPS 的吞吐率在高负载时不理想），某条流的部分信元可能会被快速交换到平面的输出端，而部分信元可能会被延迟在中间层平面的输入队列中；当那些被快速交换的信元到达复用器中 VIQ 时，会出现比较严重的乱序；并且当端口数增加、输入流量模型的突发长度越大或目的端口的非均匀分布程度更高时，乱序的程度会更高。

本文考虑是否能在同时使用 CIOQ 结构作为中间层平面和使用分布式调度算法的情况下，降低复用器中所需要的高速缓存的数量，从而使得整个 PPS 更加可行，易于实现。本文提出的 PDPPS（practical distributed parallel packet switch）结构，它是以输出端口带 VIQ 的 CIOQ 结构作为中间层交换平面，并使用分布式调度算法的并行分组交换结构。各个解复用器和复用器分别使用了分布式信元分派和重组算法，实现了每条流在各个平面之间的负载均衡和流内信元的保序转发；并且每个解复用器和复用器所需要的高速缓存大小都仅为 NK 个信元，降低了成本，更易于硬件实现。

3 PDPPS 交换结构

PDPPS 的体系结构如图 2 所示，解复用调度器 DS 采用轮询方式将每条流的信元通过内部输入链路均匀分派到 K 个 CIOQ 中间层平面；中间层 CIOQ 交换平面分布独立运行，将信元交换到相应的输出队列；最后复用调度器 MS 将每条流的信元按序从输出端口输出。其中，解复用器运行于外部链路速率 R，包含了 K 个队列，用于缓存那些由于内部输入链路被占用而无法传输到中间层平面的信元；各个解复用器中的 DS 独立分布式运行，不需要与其他部件通信。中间层平面为输入端口带 VOQ 队列、输出端口带 VIQ 队列的 CIOQ 交换结构。复用器运行于外部链路速率 R；包含了 K 块缓存，每块缓存被分为 N 个队列，每个队列的大小为 1，用于缓存从中间层平面的 VIQ 队列传输过来的队头信元；各个复用器中的 MS 独立分布式运行，不需要与其他部件通信。为了保证 MS 在进行输出调度时，当前

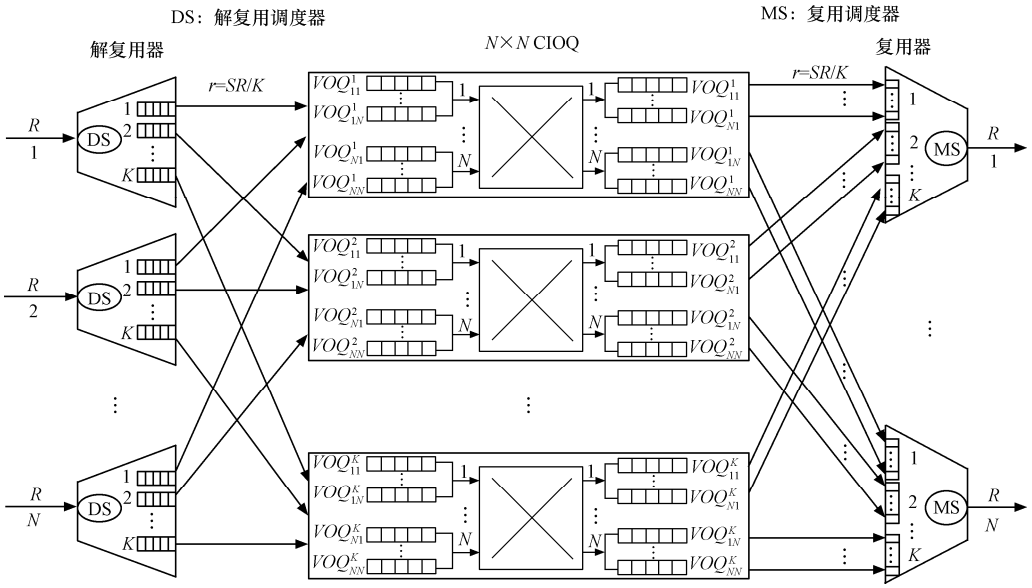


图 2　PDPPS 体系结构

输出端口的 N 条流不出现内部输出链路争用的情况，这里引入了 VOL（virtual output link）；VOL 连接着中间层平面的 VIQ 队列和解复用器中相应的长度为 1 的队列。

为了后面描述的方便，这里给出相关结构和队列的定义。

D_i：连接第 i 个输入端口的解复用器；

M_j：连接第 j 个输出端口的复用器；

S_k：第 k 个中间层交换平面；

PQ_k^i：位于解复用器 D_i 中的队列，用于缓存因相应内部输入链路被占用而无法发送到平面 S_k 的信元；

$SVOQ_{ij}^k$：位于平面 S_k 的第 i 个输入端并用于缓存流 $flow(i,j)$的虚拟输出队列；

$SVIQ_{ij}^k$：位于平面 S_k 的第 j 个输出端并用于缓存流 $flow(i,j)$的虚拟输入队列；

$MVIQ_{ki}^j$：位于复用器 M_j 中的队列，长度为 1，用于缓存从平面 S_k 的队列 $SVIQ_{ij}^k$ 的队头传送过来的信元；

$VOL(i,j,k)$：连接着 $SVIQ_{ij}^k$ 和 $MVIQ_{ki}^j$ 的内部虚拟输出链路。

4　PDPPS 调度算法

4.1　负载均衡的分布式信元分派算法

PDPPS 解复用器的结构如图 3 所示，包含了一个分布独立运行的调度器 DS 和 K 个长度为 N 的先进先出队列，每个队列缓存因内部输入链路被占用而无法发送到相应中间层平面的信元。解复用器 D_i 为 N 个输出端口维护 N 个独立的轮询指针 $P_i^1,\cdots,P_i^N$，指针的取值范围为$\{1,\cdots,K\}$。初始时，D_i 的所有指针 P_i^j（$1\leqslant j\leqslant N$）均被设置为 0。当一个新信元 C_{ij} 到达解复用器 D_i 时，执行如下分派操作。

步骤 1　根据 C_{ij} 的目的端口 j，获取轮询指针 P_i^j 的值。

步骤 2　将 C_{ij} 分派到指针 P_i^j 所指向的 PQ 队列中(假设 $P_i^j=k$，则将该信元分派到 PQ_k^i)。

步骤 3　将指针 P_i^j 的值更新为(P_i^j+1)%K。

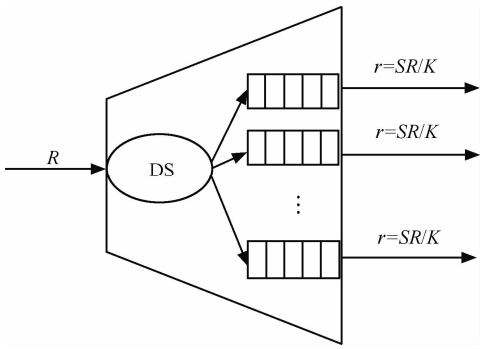


图 3　解复用器的结构

由于算法针对每条流采用了轮询分派方式，可以保证每条流的数据在各个平面之间的负载均衡，从而提高了各个平面的资源利用率，有利于提高系统的整体吞吐率，降低交换的平均时延。

根据文献[13]，解复用器中队列的长度为 N 可以保证不会发生缓存溢出。每个解复用器维护一个运行于外部链路速率 R，大小为 NK 的缓存。以端口数目 N=256，信元长度为 64byte，交换平面数目 K=10 为例，缓冲区大小为 1.28Mbyte；采用目前的 SRAM 技术可将该缓冲块集成在芯片上。

4.2 中间层交换平面

PDPPS 的中间层平面使用了输出端口带 VIQ 的 CIOQ 交换结构。采用这种结构的好处是，通过在中间层平面的输出端为每条流设置大量低速 VIQ 队列，取消了原本需要在复用器中维护的大量高速缓存，相当于将 VIQ 队列从运行于线速率 R 的复用器移到了运行于内部链路速率 r 的中间层平面输出端中，降低了硬件成本，提高了方案的可行性。

但这样做的问题是，复用器在进行输出调度时，受到内部输出链路运行速率的限制，会出现同一个输出端口的不同流争用同一条内部输出链路的现象（即同一复用器的不同流从同一中间层平面读取下一个输出信元）。因此，在 PDPPS 的中间层平面和复用器之间，使用了虚拟输出链路 VOL 来避免链路争用，从而降低整个系统的平均时延。

4.3 分布式信元保序重组算法

PDPPS 复用器的结构如图 4 所示，包含了一个分布独立运行的调度器 MS 和 K 块缓存，每块缓存存储从相应中间层平面发送过来的信元，每块缓存被分为 N 个长度为 1 的队列 MVIQ，用于存储每条流下一个可以发送的信元。复用器 M_j 为 N 条流维护 N 个独立的轮询指针 $P_j^1,\cdots,P_j^N$，指针的取值范围为{1,…,K}。M_j 还维护了一个流指针 I_j，取值范围为{1,…,N}，用来表示下一次调度时首先尝试输出的流。设置流指针 I_j 的目的是为了让属于同一个输出端口的 N 条流公平地获得输出机会。初始时，所有指针 P_j^i（$1\leqslant i\leqslant N$）和 I_j 都被设置为 0。每个复用器独立地执行以下重组算法。

步骤 1 从流指针 I_j 所指向的流开始遍历目的端口为 j 的所有 N 个流；对于其中任意一个流 $flow(i,j)$，执行步骤 2。

步骤 2 根据 $flow(i,j)$的输入端口号 i 读取 M_j 的轮询指针 P_j^i 的值，假设为 k。

步骤 3 根据 i 值和 k 值读取 M_j 的队列 $MVIQ_{ki}^j$。若队列 $MVIQ_{ki}^j$ 非空，则执行步骤 4；否则，返回步骤 1 尝试输出流 $flow(i+1,j)$的信元。

步骤 4 输出 $MVIQ_{ki}^j$ 的队头信元，并且更新轮询指针 P_j^i 的值为(P_j^i +1)%K，更新流指针 I_j 的值为(I_j +1)%N；完成当前调度。

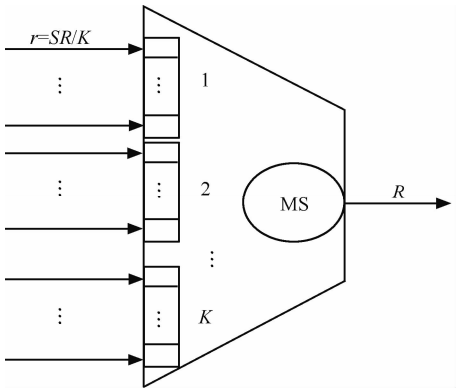


图 4 复用器的结构

每个复用器维护一个运行于外部链路速率 R，大小为 NK 的缓冲区。以端口数目 N=256，信元长度为 64byte，交换平面数目 K=10 为例，所需缓冲区大小为 1.28Mb，采用目前的 SRAM 技术可将该缓冲区集成在芯片上。

定理 1 PDPPS 能够保证每条流按序输出。

证明 由于 PDPPS 所使用的信元分派算法和重组算法都属于轮询算法，且起始点都是第一个中间层平面；解复用器将每条流的信元循环分派到各个平面；而复用器也是循环从各个平面读取每条流的信元。因此只需要证明每一条流在经过任意一个平面的交换过程具有保序性，就可以说明 PDPPS 能够保证每条流的按序输出。任意一条流在被 PDPPS 交换时所经过的缓存队列，包括了 PQ_k^i、$SVOQ_{ij}^k$、$SVIQ_{ij}^k$ 和 $MVIQ_{ki}^j$。由于这 4 个队列都是 FCFS 机制，因此任意一条流在这条路径上一定是按顺序从解复用器传输到复用器，每一条流在经过任意一个平面的交换过程具有保序性。

定义 1 只要复用器的 MVIQ 队列或中间层平面的输出端存在流 $flow(i,j)$的下一个输出信元，复用器 M_j 就会处于“忙”状态，那么称复用器 M_j 具有 work-conserving 特性。

定理 2 在 PDPPS 的复用器中，设置大小为 NK 的高速缓存可以保证复用器具有 work-conserving 特性。

证明 如果复用器 M_j 的队列 $MVIQ_{ki}^j$ 中存在流 $flow(i,j)$的下一个应该输出的信元，那么 M_j 可以在下一个时隙输出该信元，定理 2 显然成立。

当复用器 M_j 输出了队列 $MVIQ_{ki}^j$ 中的信元后，

如果中间层平面 S_k 的输出端存在流 $flow(i,j)$的下一个应该输出信元，根据定理 1 可知，$flow(i,j)$在平面 S_k 中的最老信元一定位于平面 S_k 的输出端队列 $SVIQ_{ij}^k$ 头部；队列 $SVIQ_{ij}^k$ 将其队头信元通过 $VOL(i,j,k)$传送到 $MVIQ_{ki}^j$ 的时间为 KT/S（T 为一个外部时隙）；而根据上述的信元重组算法，复用器 M_j 至少需要经过 KT 的时间才会再次调度 $MVIQ_{ki}^j$；由于 $S\geqslant 1$，因此 $KT/S\leqslant KT$，即在 $MVIQ_{ki}^j$ 被复用器 M_j 再次调度输出之前，$flow(i,j)$的下一个信元已经到达 $MVIQ_{ki}^j$，因此 M_j 可以保持“忙”状态，定理 2 成立。

4.4　理论性能分析

这里主要通过与一个使用 FCFS 队列的 OQ 交换结构比较，来对使用 2 倍加速的 iSLIP 作为中间层平面调度算法的 PDPPS 的性能进行分析。

定理 3　使用 2 倍加速的 iSLIP 作为中间层平面调度算法的 PDPPS 能够模仿一个 FCFS-OQ 交换结构（时延差在 $3N$ 个内部时隙的有限范围内）。

证明　一个信元被 PDPPS 传输时所经历的排队时延主要由以下 3 个部分组成。

1) 在被传输到中间层交换平面之前，信元可能会被缓存在解复用器的 PQ 队列中，因为 PQ 队列的长度为 N，并且每隔一个内部时隙至少有一个信元通过内部输入链路被传输到中间层平面，所以信元在解复用器中最多被缓存 N 个内部时隙。

2) 由于使用 2 倍加速的 iSLIP 作为调度算法的 CIOQ 结构可以模仿一个 FCFS-OQ 结构[4]，因此信元在中间层平面中经历的时延与在 FCFS-OQ 中经历的一样。

3) 出现信元在复用器中被缓存最长时间的情况是：对于信元 C_{ij}，假设其在外部时隙 T_1 结束时从中间层平面 S_k 传输到达复用器 M_j 的队列 $MVIQ_{ki}^j$，而调度器 MS_j 在 T_1 刚从队列 $MVIQ_{ki}^j$ 调度输出了一个信元，那么只有当队列 $MVIQ_{ki}^j$ 下一次被 MS_j 调度时，信元 C_{ij} 才能被输出；根据上述的信元重组算法可知，当 MS_j 将其他 $NK-1$ 个 MVIQ 都尝试调度了一遍之后，队列 $MVIQ_{ki}^j$ 才会获得再一次被调度的机会；在 MS_j 尝试调度其他 $NK-1$ 个 MVIQ 的过程中，如果每个 MVIQ 都有信元需要被输出，那么由于每个外部时隙 MS_j 只能调度输出一个信元，因此至少经过 $NK-1$ 个外部时隙，$MVIQ_{ki}^j$ 才会再次被调度，信元 C_{ij} 才能被输出；加上 T_1 这一个外部时隙，信元 C_{ij} 被缓存的最长时间为 $(NK-1)+1=NK$ 个外部时隙；假设外部时隙的大小为 T，内部时隙大小为 t，在加速 $S=2$ 的情况下，$t=KT/2$，NK 个外部时隙=$NKT=2Nt$，即 $2N$ 个内部时隙。

所以，相比较于 FCFS-OQ 结构，一个信元在 PDPPS 中所经历的相对时延的最大值为 $N+2N$ 个内部时隙，即 $3N$ 个内部时隙。

5　性能模拟与分析

5.1　实验环境及相关参数

笔者使用C++面向对象技术开发了模拟仿真交换环境。开发的模型包括 OQ、iSLIP IQ、集中式 PPS、VIQ PPS、IOQ PPS 和 PDPPS。其中，IQ 结构所使用的 iSLIP 调度算法、输入流量模型中的贝努利模型和突发模型直接借鉴了 SIM 模拟器[19]。所使用的输入流量模型为：1）贝努利一致均匀流；2）贝努利一致非均匀流；3）突发均匀流；4）突发非均匀流。突发业务到达过程通过 ON-OFF 模型产生，突发长度服从几何分布，平均突发长度为 32。非均匀流的目的端口分布服从 $\lambda_{i,i}=0.67\rho$、$\lambda_{i,i+1}=0.33\rho$ 分布（ρ 为业务负载）。

在 4 种流量模型下对各种结构和 PDPPS 的时延性能进行了实验仿真。这里，把使用 2 倍加速 iSLIP 作为中间层平面调度算法的 PDPPS 记为 2iSLIP-PDPPS。在实验过程中，集中式 PPS 的内部链路使用了 2 倍的内部加速，IOQ PPS 和 2iSLIP-PDPPS 的内部链路和中间层平面都使用了 2 倍的内部加速。在所有的实验中，模拟时间为 1 000 000 个外部时隙，交换结构的规模为 32×32；所有 PPS 结构的中间层平面的数目为 10；中间层平面的 VOQ 和 VIQ 队列的长度都设为无限。

5.2　仿真结果分析

图 5 给出了在贝努利一致均匀流量模型下各种交换结构的性能。在低负载时，iSLIP 的性能好于所有 PPS 结构；当负载高于 0.96 后，PPS 结构的性能明显优于 iSLIP。这是因为在低负载时，PPS 结构的时延组成中，内部链路的传输时延和流内信元的重排序时延占较大比重，而在高负载时比重变小。相反使用极大匹配的 iSLIP 算法在

高负载时匹配的效果受到很大影响，大量信元在输入队列中排队等待。在各种负载下，2iSLIP-PDPPS 的时延都小于 VIQ PPS。这主要包括两方面原因，一方面是因为 2iSLIP-PDPPS 的内部输入链路和内部输出链路都使用了 2 倍的链路加速，所以 2iSLIP-PDPPS 的内部链路传输时延比 VIQ PPS 小了 10/2+10/2=10 个外部时隙；另一方面是因为在 VIQ PPS 中，受内部输出链路约束的影响，信元可能会被阻塞在中间层平面的输出端，增加了信元的等待时延，而 2iSLIP-PDPPS 由于在中间层平面和复用器之间引入了虚拟输出链路 VOL，消除了信元在中间层平面输出端的阻塞，降低了信元的交换时延。在负载低于 0.96 时，集中式 PPS 的平均时延比 2iSLIP-PDPPS 低 40%左右，这主要是因为集中式 PPS 使用了 OQ 结构作为中间层平面，消除了信元在中间层平面的输入端排队时延；但 OQ 结构存在的 N 倍加速问题严重影响了交换结构的可扩展性。在各种负载下，IOQ PPS 的平均时延只比 2iSLIP-PDPPS 低 5%左右；但相对于使用集中式调度算法的 IOQ PPS，2iSLIP-PDPPS 采用了分布式调度算法，各个部件完全独立运行，消除了系统的通信开销，极大降低了硬件实现成本。在各种负载下，2iSLIP-PDPPS 的时延都比 OQ 结构大，但时延相差的大小都在一个固定的范围内。这也就进一步验证了定理 3，说明 2iSLIP-PDPPS 可以模仿一个 FCFS-OQ 交换结构（最多相差 3N 个内部时隙）。图 6 给出了在贝努利一致非均匀流量模型下各种交换结构的性能。2iSLIP-PDPPS 和 iSLIP、集中式 PPS、IOQ PPS、VIQ PPS、OQ 的时延性能关系与图 5 类似。

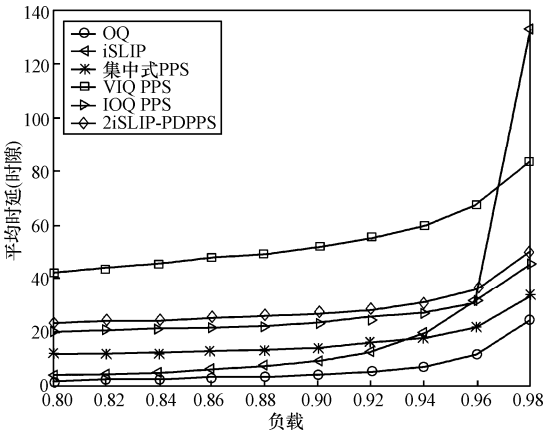


图 5　贝努利一致均匀流量模型下各种交换结构的性能

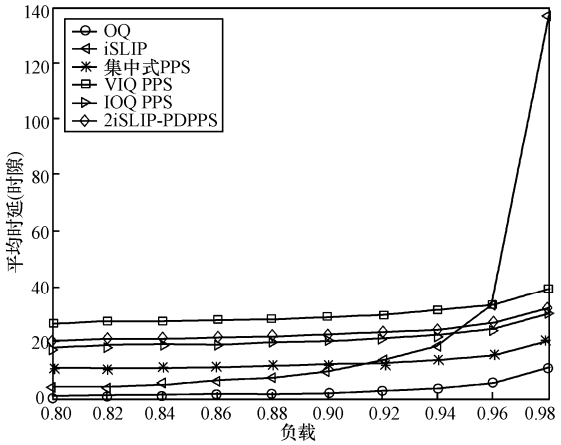


图 6　贝努利一致非均匀流量模型下各种交换结构的性能

图 7 给出了在突发均匀流量模型下各种交换结构的性能。当负载大于 0.7 时，各种 PPS 的时延性能都优于 iSLIP；原因与贝努利一致均匀流下的情况一样。在各种负载下，IOQ PPS 的平均时延性能只比 2iSLIP-PDPPS 高 2%左右；但 IOQ PPS 却需要使用集中式调度算法，所有中间层平面需要协调同步运行，各个部件都需要和集中式调度器通信，需要大量的通信开销，增加了硬件实现的复杂度。同样，2iSLIP-PDPPS 的平均时延总是低于 VIQ PPS，高于集中式 PPS 和 OQ；只是这个时延的差值在负载较高时会有一定的上下浮动，这是由于负载越大，突发的随机性对交换结构的性能影响越大，性能的浮动也越大。当负载达到 0.92 以上时，iSLIP 已经变得不稳定；当负载达到 0.96 以上时，所有交换结构的性能都变得不稳定。可见真实网络流量的突发特性对交换结构的性能有着极大的影响。图 8 给出了在突发非均匀流量模型下各种交换结构的性能。当负载大于 0.8 时，各种 PPS 的时延性能都

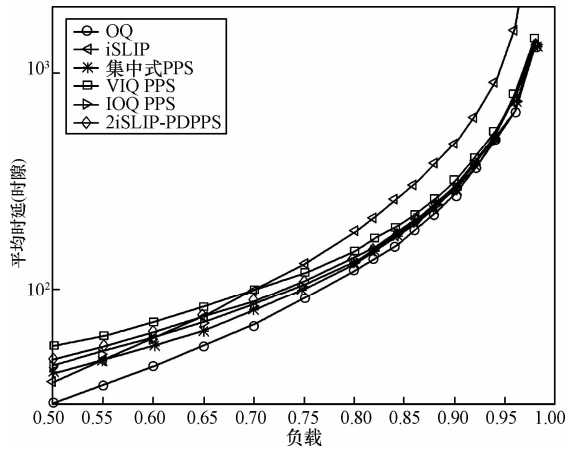


图 7　突发均匀流量模型下各种交换结构的性能

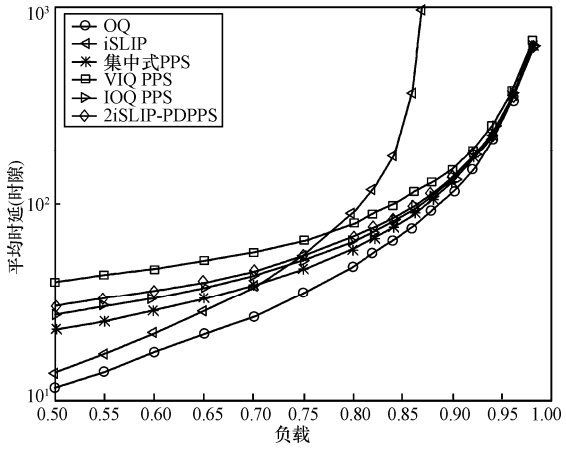


图 8　突发非均匀流量模型下各种交换结构的性能

优于 iSLIP。当负载大于 0.86 时，iSLIP 的性能变得非常不稳定；可见突发非均匀流对于使用 iSLIP 算法的 IQ 结构的性能影响比较严重。然而，各种 PPS 的性能并未受太大影响，原因是 IOQ PPS 和 2iSLIP-PDPPS 的中间层平面使用了 2 倍加速的 iSLIP 调度算法，其他 PPS 的中间层平面使用的是 OQ 结构，这两种平面对输入的非均匀分布特性并不敏感。同样，2iSLIP-PDPPS 的时延总是低于 VIQ PPS，略微高于 IOQ PPS 和集中式 PPS。

6　结束语

本文提出了一种可行的分布式并行分组交换结构。从理论上证明了在分布式信元分派算法和重组算法的协同工作下，PDPPS 具有保序性。整个交换结构中，由于使用了不存在 N 倍加速问题的 CIOQ 结构作为中间层平面，并且解复用器和复用器中都只需要设置大小为 NK 个信元的高速缓存，因此降低了硬件实现成本，提高了方案的可行性。仿真实验结果表明，无论是在贝努利流量模型还是在突发流量模型下，PDPPS 的平均时延性能都优于目前主流的 VIQ PPS，略微低于使用集中式调度算法的集中式 PPS 和 IOQ PPS，但 PDPPS 在算法复杂度、通信开销和需要的高速缓存数量等方面都优于集中式 PPS 和 IOQ PPS。下一步的工作主要包括提供服务质量保证和多播等方面的支持。

参考文献：

[1] MCKEOWN N, IZZARD M, MEKKITTIKUL A, *et al*. The tiny tera: a packet switch core[J]. IEEE Micro, 1997, 17(1):26-33.

[2] Digital Equipment Corporation. GIGAswitch[EB/OL]. http://www.networks.digital.com, 1997.

[3] Ascend Communications. GRF family of switches[EB/OL]. http://www. ascend.com, 1998.

[4] MCKEOWN N. The iSLIP scheduling algorithm for input-queued switches[J]. IEEE/ACM Transactions on Networking, 1999, 7(2): 188-201.

[5] SERPANOS D N, ANTONIADIS P I. FIRM: a class of distributed scheduling algorithms for high-speed ATM switches with multiple input queues[A]. Proceedings of IEEE INFOCOM'00[C]. Tel Aviv, Israel, 2000. 548-554.

[6] CHAO H J. Satur: a terabit packet switch using dual round-robin[J]. IEEE Communications Magazine, 2000, 38(12): 78-84.

[7] MCKEOWN N, MEKKITTIKUL A, ANANTHARAM V, *et al*. Achieving 100% throughput in an input-queued switch[J]. IEEE Transactions on Communications, 1999, 47(8):1260-1267.

[8] LI Y, PANWAR S, CHAO H J. Frame-based matching algorithms for optical switches[A]. Proceedings of IEEE Workshop on High Performance Switching and Routing (HPSR 2003)[C]. Torino, Italy, 2003. 97-102.

[9] OKI E, ROJAS C R, CHAO H J. A pipeline-based maximal-sized matching scheme for high-speed input-buffered switches[J]. IEEE Transactions on Communications, 2002, 85(7):1302-1311.

[10] ATTYIA H, HAY D. Randomization does not reduce the average delay in parallel packet switches[J]. Society for Industrial and Applied Mathematics, 2008, 37(5):1613-1636.

[11] YANG F, WANG Z K, CHEN J Y, *et al*. A parallel packet switch supporting differentiated QoS based on weighted layer assignment[A]. Proceedings of the 5th International Conference on Wireless communications, networking and mobile computing[C]. Beijing, China, 2009. 4286-4289.

[12] DUNCANSON J. Inverse multiplexing[J]. IEEE Commun Mag, 1994, 32:34–41.

[13] IYER S, AWADALLAH A, MCKEOWN N. Analysis of a packet switch with memories running slower than the line rate[A]. Proceedings of the IEEE INFOCOM'00[C]. Israel, 2000. 529-537.

[14] IYER S, MCKEOWN N. Making parallel packet switches practical[A]. Proceedings of the IEEE INFOCOM[C]. Alaska, USA, 2001. 1680-1687.

[15] ASLAM A, CHRISTENSEN K J. A parallel packet switch with multiplexors containing virtual input queues[J]. Computer Communications, 2004,27:1248-1263.

[16] MNEIMNEH S, SHARMA V, SIU K. On scheduling using parallel input-output queued crossbar switches with no speedup[A]. Proceedings of IEEE Workshop on High Performance Switching and Routing[C]. Texas, USA, 2001. 317-323.

[17] 戴艺, 苏金树, 孙志刚. 一种维序的基于组合输入输出排队的并行交换结构[J]. 软件学报, 2008, 19(12):3207-3217.

（下转第 32 页）