一种面向片上网络的链路调度算法

张 哲 龙 翔 高小鹏

(北京航空航天大学 计算机学院, 北京 100191)

摘 要: 新兴的片上网络(NoC, Network-on-Chip)通常采用虫孔交换技术,其中的链路调度机制难以保证报文级的转发延迟.提出的逆向锚点轮转(RARR, Reverse Anchored Round-Robin)调度算法结合了逐个微片轮转(FFRR, Flit-by-Flit Round-Robin)和逐个报文轮转(PPRR, Packet-by-Packet Round-Robin)调度算法的特点. RARR算法在报文的头微片抵达目的节点前以逐个微片的方式实施调度;此后以最后一跳的链路为起点,沿该报文的转发路径逆向的、逐跳的为所有片段请求和调度锚点. RARR算法将获得锚点的报文设置为最高优先级,对其实施报文级的调度;当锚点报文转发过程中断时,以逐个微片的方式轮转调度其他报文. RARR算法的基本思想源于锚点轮转(ARR, Anchored Round-Robin)调度算法,但是其中关键的锚点调度机制更具确定性,同时消除了ARR算法中的死锁问题.利用周期精确的虫孔交换网络仿真环境量化评估了常见的轮转调度算法,包括FFRR, PPRR, ARR和RARR.实验结果表明, RARR算法具有最优的性能.

关 键 词: 片上网络; 虫孔交换; 链路调度; 轮转

中图分类号: TP 393

文献标识码: A 文章编号: 1001-5965(2008)08-0930-05

Link scheduler for network-on-chip

Zhang Zhe Long Xiang Gao Xiaopeng

(School of Computer Science and Technology, Beijing University of Aeronautics and Astronautics, Beijing 100191, China)

Abstract Wormhole-switching is usually employed in the emerging network-on-chip (NoC), in which the link scheduler can hardly guarantee the packet-level latency. Reverse anchored round-robin (RARR) is proposed as hybrid of the flit-by-flit round-robin (FFRR) and the packet-by-packet round-robin (PPRR). In the scheme of RARR, before the head flits have arrived at the destination, the packets are forwarded flit by flit. Then the scheduler at the destination link starts attempting to mark all fragments of the packet as anchors, which is accomplished by requesting and scheduling following the path reversely and hop by hop. The anchored packet takes priority over others, and will be scheduled at packet level. Others will be scheduled flit by flit only if the anchored one breaks. The RARR is inspired by the anchored round-robin (ARR), but employs more determinate anchoring scheme and eliminates the deadlock in ARR. Familiar round-robin were quantified via a cycle-accurate wormhole network simulator, including FFRR, PPRR, ARR and RARR. The RARR was shown to be most efficient among them.

Key words network-on-chip; wormhole switching; link scheduling; round-robin

近年来,随着多核化(multi-core)成为芯片设计的重要发展趋势,基于分组的互连网络正逐步取代传统的总线互连方式,提供性能更高、伸缩性更好的内核间通信服务,被称为片上网络(NoC, Network-on-Chip)[1]. NoC源于并行计算机网络,由于实现环境所限而面临更加严格的资源成本约束.其中,由于缓冲区在片上路由器的实现成本中占有较大的比重[2],因此缓冲资源受限成为NoC设计的重要特征之一.

目前,低延迟、低成本的虫孔(wormhole)交换

收稿日期: 2007-09-07

作者简介: 张 哲(1978-),男,河北深县人,博士生, z zhang@ cse buaa edu cn.

技术以及虚通道 (virtual-channel) 流控技术被广泛应用于 NoC 领域中. 在虫孔交换网络中, 基于微片的流控方式在降低转发延迟和实现成本的同时, 也对链路调度机制产生了严重的负面影响, 使其难以保证报文级的转发延迟. 在 NoC 中, 链路调度机制的有效性由于缓冲区尺寸进一步缩小而面临更加严峻的挑战.

本文以缓冲资源受限的片上虫孔交换网络为背景, 提出了逆向锚点轮转 (RARR, Reverse Anchored Round-Robin) 调度算法. 该算法借鉴了文献[3]提出的锚点轮转 (ARR, Anchored Round-Robin) 调度算法, 并且改进了其中的锚点调度过程: RARR 算法在报文头微片抵达目的节点后才开始逐跳的请求和调度锚点, 不但具有更好的确定性, 并且消除了 ARR 算法中存在的死锁问题. 因此, RARR 算法能够更有效的为报文建立独占式的转发通路, 降低了报文转发的平均延迟.

1　虫孔交换网络中的链路调度算法

在虫孔交换网络中, 各个虚通道的报文通过复用物理链路完成转发. 因此链路调度算法决定了报文实际占用的带宽, 是影响网络性能的重要因素[4]. 本节分析了缓冲区受限的虫孔交换方式给链路调度带来的挑战, 介绍了已有的相关研究成果及其存在的不足.

1.1　面临的挑战

在传统的存储-转发 (SAF, Store-And-Forward) 网络中, 流控和链路调度的单位均为报文, 在流控条件满足的情况下, 调度算法可以根据报文长度确定其经过本跳的延迟.

在虫孔交换网络中, 由于流控单位是微片, 而且下游转发通路可能出现阻塞的情况, 因此流控条件完备只能保证若干微片完成转发. 同时, 虫孔交换网络中的报文转发过程具有分布式存储、流水式转发的特点, 由于各个路由器中的链路复用情况彼此独立, 各段流水线容易出现转发速率不匹配的情况和流水线气泡 (bubble), 从而导致转发过程中断. 可见, 虫孔交换网络中的链路调度存在显著的不确定性, 调度算法难以保证报文级的转发延迟.

1.2　相关工作

并行计算机网络和 NoC 领域的相关研究借鉴了广域网络研究中的基本思想, 提出了先到先服务 (FCFS, First Come First Served) 算法[5]、最近最少服务 (LRS, Least Recently Served) 算法[6]、基于 FCFS 方式的前瞻 (LA, Look-Ahead) 调度算法[4]、最老报文优先 (oldest-first) 调度算法[7]等. 其中, FCFS, LRS 等时限相关的调度算法需要大量的触发器和逻辑运算单元维护虚通道的状态, 增加了调度算法和路由器的实现成本; LA 算法还需要对路由算法等进行约束.

轮转是另一类主要的调度方法, 具有显著的成本优势, 更能适应资源约束比较严格的 NoC 的需求, 因此也得到了广泛的应用. 常用的轮转调度算法如下:

1) FFRR 算法[8-9]以轮转方式调度各个虚通道, 每轮调度仅为当前虚通道转发一个微片. 该算法提供了良好的公平性, 但是也导致了报文转发延迟的普遍增加.

2) PPRR 算法[10]在当前调度的报文由于阻塞或气泡中断时调度下一报文. 该算法在理想情况 (报文能够连续转发) 下能够克服 FFRR 算法的性能问题. 但是在虫孔交换网络和 NoC 中, 分布存储的报文由于阻塞和气泡难以保证连续转发, 导致其性能也难以达到预期的水平.

3) ARR 算法[3]结合了 FFRR 和 PPRR 算法的特点: 各个报文片段独立请求链路的锚点, 获得锚点的报文将优先占用链路, 其他报文只能在锚点报文转发中断时获得逐个微片的轮转调度. ARR 算法借助锚点实现连续的报文转发, 其有效性决定于锚点调度的同步. 由于各段链路的锚点调度彼此独立, 以及下游通路状态无法预测, ARR 算法的有效性难以得到保证, 报文在最坏情况下只能以 FFRR 方式完成转发. 同时, ARR 算法中的锚点调度机制存在死锁. 例如: 当两个报文同时请求两段链路的锚点时, 二者可能分别占有一个锚点而进入死锁状态.

在 NoC 中, 报文由于缓冲有限而跨越更多的链路, 因此 PPRR 和 ARR 算法更加难以通过独立的分布式调度方式实现报文级的转发. 同时, 链路状况随着负载的增加而更加复杂, 二者的有效性也随之降低.

2　逆向锚点轮转调度算法

报文转发的连续性是决定轮转调度算法性能的重要因素, ARR 算法中的锚点协同方式给出了一种可行的解决途径. 本文提出的 RARR 算法借鉴了上述思想, 并且改进了其中关键的锚点调度机制.

2.1　基本原理

虫孔交换网络中的报文转发过程可以划分为

以下两个阶段: ①头微片注入网络到头微片抵达目的节点; ②头微片抵达目的节点到尾微片抵达目的节点. 二者的区别在于是否存在阻塞. 在第②阶段, 目的节点可以视为理想的下游通路, 如果报文的各个片断能够同时独占相应链路, 报文将可以保持连续的转发.

RARR算法在第①阶段采用 FFRR 方式实施调度, 不但消除了锚点请求的盲目性, 而且保证了调度的公平性. 在第②阶段, RARR 算法在报文头微片抵达目的节点并且占有最后一段链路的锚点后, 沿报文的转发通路逆向建立独占通路, 从而增加了锚点调度的确定性. 其中, 锚点请求沿转发通路逆向逐跳进行, 直到当前片段包含尾微片; 报文只有获得后一跳 (接近目的节点) 链路的锚点后才可以请求前一跳 (接近源节点) 链路的锚点. 与 ARR 算法一样, RARR算法在各段链路中优先转发锚点报文, 当无锚点报文或者锚点报文的转发过程中断时, 以 FFRR 方式调度其他报文.

图 1给出了 2个路径重叠的报文在 RARR 算法调度下完成转发的过程, 包括: 逆向、逐跳的调度锚点, 建立独占的转发通路, 以及实现报文级的转发.

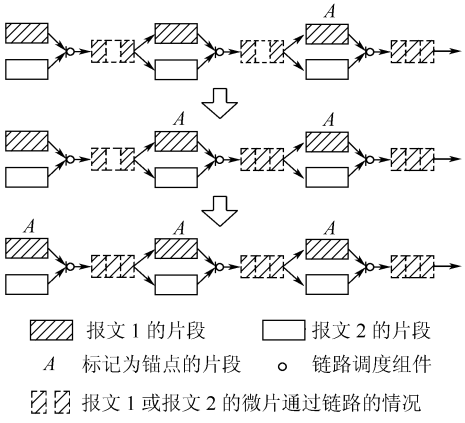


图 1 RARR调度下的报文转发过程

RARR 算法中的逐跳调度方式消除了 ARR 算法中存在的死锁风险, 能够确保一个以上的报文在进入第②阶段后实现连续的转发. 下面的定理 1和推论 1说明了 RARR算法的上述特点.

定理 **1** RARR 算法中的锚点请求无死锁, 当且仅当路由算法无死锁.

证明 网络中存在的报文转发路径可以表示为通道相关图 $D = G(C, E)$. 其中, 顶点 C 表示网络中的链路, 有向边 E 表示路由函数可以在两条链路之间建立直接的转发路径. 根据文献 [11]中的定理 1, 路由算法无死锁的充分必要条件是 D 中不存在环路.

RARR算法中的锚点请求是路由算法的反函数, 其通道相关图 $D' = G(C, E')$与 D 中的顶点相同, 边的方向相反. 显然, D'中存在环路的充分必要条件是 D 中存在环路. 同时, 根据文献 [11]中定理 1的证明, 锚点请求无死锁的充分必要条件是 D'中不存在环路. 因此锚点请求无死锁的充分必要条件是路由算法无死锁. 证毕

推论 **1** 当路由算法无死锁时, RARR 算法可以保证至少一个发出请求的报文获得所需的全部锚点.

证明 定义有向图 $D'' = G(C'', E'')$, 其中 C''为所有请求锚点的报文集合, E''表示 C''中的二元关系 R: $c''_i R c''_j$表示报文 c''_i和 c''_j的锚点请求在某段链路 L_{ij}发生竞争, 同时 c''_i最终获得了锚点调度, 称为 c''_i抢占了 c''_j的锚点. D''中不存在环路, 因为:

1) 如果 $c''_i R c''_i$, 则锚点请求产生的通道相关图 D'中存在环路, 根据上述定理 1的证明, 路由算法存在死锁, 与推论条件矛盾.

2) 如果 $c''_i R c''_j R c''_k$, 即 c''_i在 L_{ij}抢占了 c''_j的锚点, c''_j在 L_{jk}抢占了 c''_k的锚点, 那么通道相关图 D 和 D'中存在 L_{jk}和 L_{ij}之间的通路, 因此如果 c''_k通过 L_{ij}并且获得 L_{jk}的锚点就会与 c''_i竞争 L_{ij}的锚点, 即可能出现 $c''_i R c''_k$. 因此, 如果存在 $c''_i R \cdots R c''_i$, 则可能存在 1)中的 $c''_i R c''_i$.

根据文献 [12]中的定理 9.10, 无环有向图 D 具有唯一的非空顶点基, 包含 D 中所有的无引入弧顶点. 因此, D 中存在无引入弧的顶点, 即存在不被任何其他报文抢占锚点的报文, 可以获得所需的全部锚点. 证毕

综上所述, RARR 算法是一种比较保守的链路调度算法, 通过改善报文的第②阶段转发延迟提高性能. 因此其优势在第②阶段转发延迟占报文转发延迟的相当比重时才比较显著. 直观地看, 就是报文跨越的路由器数目与转发所需的跳数可比. 在 NoC 中, 缓冲受限导致报文跨越的路由器数目较多; 同时, 网络通信本身具有显著的局部性特征, 而且目前的多核处理器和 NoC 研究在专用领域更加活跃[13], 定制设计方式将使内核间通信的局部性更加显著, 报文转发所需的跳数更加有限. 因此, RARR 算法是一种适合 NoC 领域的链路调度算法.

2.2 算法实现

RARR算法是一种分布式的链路调度算法, 各个路由器和链路的调度组件通过锚点请求信号实现协同. 图 2给出了调度组件的状态机, 其实现

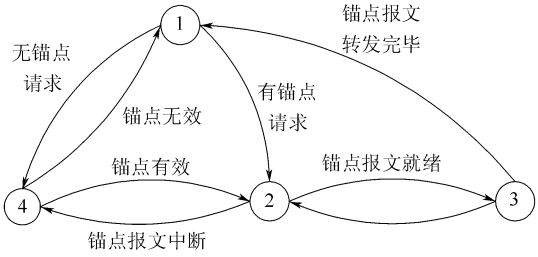


图 2　RARR调度组件的状态机

一轮调度的伪代码如下:

```
if (Anchor >= 0) { //锚点有效,即锚点被占用
  if (FlowValid(Anchor)) { //转发锚点报文
    TransOneFlit(Anchor);
    If (EndFlitTrans)
      Anchor= -1;
    If (EndFlitInVC(Anchor))
      ReseUpAnchor(Anchor);
  }
  else { //转发其他报文
    CurrVC= RR(ValidVC, LastVC);
    TransOneFlit(CurrVC);
    LastVC= CurrVC;
  }
}
else { //锚点无效
    if (DownAnchor! = 0) { //锚点请求有效
      Anchor= RR(DownAnchor, LastAnchor);
      LastAnchor= Anchor;
      if (! EndFlitInVC(Anchor))
        SetUpAnchor(Anchor);
    }
    else { //无锚点请求
      CurrVC= RR(ValidVC, LastVC);
      TransOneFlit(CurrVC);
      LastVC= CurrVC;
    }
}
```

其中, Anchor标记了占用锚点的虚通道 (报文), 初始状态为 -1(无效); FlowValid()用于确认指定虚通道的流控状态是否合法; LastVC和 CurrVC记录了上一轮和当前获得调度的虚通道; ValidVC表示流控状态合法的虚通道集合; TransOneFlit()调度指定虚通道转发一个微片; EndFlitTrans表示当前转发完成的是尾微片; DownAnchor和 Anchor是来自下游和发往上游链路的锚点请求; LastAnchor标记了上次获得锚点的虚通道; EndFlitInVC()确认指定虚通道是否包含尾微片, 调度算法据此确定是否继续向前一跳路由器请求锚点; SetUpAnchor()/ReseUpAnchor()建立 /清除对前一跳路由器的锚点请求; RR()实现了轮转调度算法.

3　仿真实验结果及分析

本文考察了 FFRR, PPRR, ARR, RARR 等 4种轮转链路调度算法. 图 3比较了在报文长度分别为 16微片和 32微片条件下各种链路算法产生的网络延迟, 其中注入负载是指报文注入速率与链路带宽的比值, 延迟的单位是周期. 可见, RARR算法在各种报文长度和负载条件下的性能均高出 FFRR算法大约 10%以上; PPRR算法和 ARR算法的性能通常随着负载的增加逐渐下降, 在饱和点附近达到甚至低于 FFRR算法的性能, 与第 1. 2节的分析相符; RARR算法在多数情况下优于 PPRR和 ARR算法 (在报文长度为 32微片和注入负载为 80%的条件下低于 PPRR算法约 4%). 下面从报文转发的 2个阶段详细分析不同算法的性能.

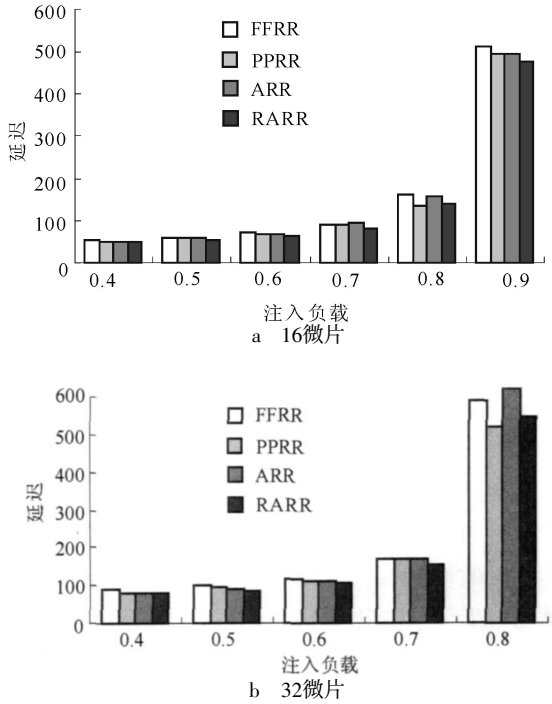
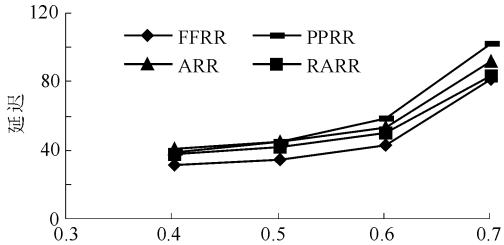


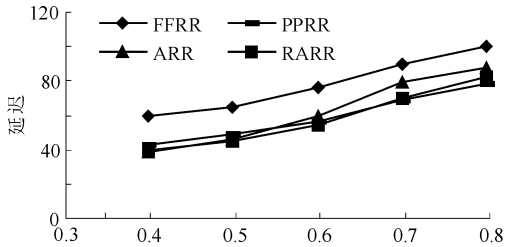
图 3　不同报文长度条件下的转发延迟

在第①阶段, FFRR算法产生的平均延迟较低 (图 4), 这是由于其他算法允许报文独占链路而阻塞其他距离目的节点很近的报文. 其中, ARR算法和 RARR算法中建立独占链路的成功率随着负载增加而降低, 二者的调度行为和性能都趋于 FFRR算法, PPRR算法则在过饱和点达到该状态. 同时, RARR算法和 PPRR算法的性能在过饱和点甚至分别超过了 FFRR算法 10%和 5%, 这

是由于高效的第②阶段调度客观上降低了网络的拥塞状况,以及 RARR 算法建立独占通路的有效性和 PPRR 算法调度的灵活性.



a 第①阶段



b 第②阶段

图 4 报文长度为 32微片条件下的2个阶段延迟(不包含过饱和点)

在第②阶段,基于微片的 FFRR 算法的性能明显低于其他基于报文的调度算法.可见,阻塞问题对调度性能的影响远远超过气泡的影响. PPRR 算法同样由于调度的灵活性使其相对于 FFRR 算法的性能优势保持在 25% 左右. ARR 算法和 RARR 算法由于依赖于独占链路建立的成功率,因此相对 FFRR算法的性能优势随着网络负载的增加而降低,其中 RARR算法降低的趋势比 ARR 算法平缓.

总的来看,随着网络负载的增加,第①阶段延迟的增长明显超过第②阶段延迟. RARR 算法在第①阶段的性能接近 FFRR 算法,虽然第②阶段的延迟略低于 PPRR 算法,但由于第①阶段延迟所占比重较大,其性能仍然超过了其他轮转调度算法.

4 结 论

本文以缓冲资源受限的 NoC为背景,提出了一种新型的轮转链路调度算法. RARR 算法结合了 FFRR 算法和 PPRR 算法的优点,保证了调度的公平性和有效性.与采用同样思想的 ARR算法相比, RARR 算法中的锚点请求和调度机制更加合理和有效,有利于在网络状况比较复杂的情况下保证调度的性能. RARR 提供了一种在缓冲受限条件下保证报文转发连续性的方法,可以与其他调度算法相结合,进一步提高网络的性能.

参考文献 (References)

[1] Benini L, Micheli G D. Networks on chips: a new SoC paradigm [J]. IEEE Computers, 2002, 35(1): 70-78

[2] Hu J, Marculescu R. Application-specific buffer space allocation for networks-on-chip router design[C]//Proceedings of IEEE/ACM International Conference on Computer-Aided Design. Washington, DC: IEEE CS, 2004: 354-361

[3] Sethu H, Shi H, Kanhere S S, et al. A round-robin scheduling strategy for reduced delays in wormhole switches with virtual lanes[C]//Proceedings of the International Conference on Communications in Computing. Las Vegas: [s.n.], 2000: 275-278

[4] Pirvu M, Bhuyan L, Ni N. The impact of link arbitration on switch performance[C]//Proceedings of 5th International Symposium on High-Performance Computer Architecture. Washington, DC: IEEE CS, 1999: 228-235

[5] Galles M. Spider: a high-speed network interconnect[J]. IEEE Micro, 1997, 17(1): 34-39

[6] Stunkel C B, Shea D G, Abali B, et al. The SP2 high-performance switch[J]. IBM Systems Journal, 1995, 34(2): 185-204

[7] Nielsen K H. Evaluation of real-time performance models in wormhole-routed on-chip networks[D/OL]. http://www.imit.kth.se/~axel/papers/2005/MSc-nielsen.pdf

[8] Kavaldjiev N, Smit G J M. An energy-efficient network-on-chip for a heterogeneous tiled reconfigurable systems-on-chip[C]//Proceedings of Digital System Design, EUROMICRO System on (DSD'04). Washington, DC: IEEE CS, 2004: 492-498

[9] Pande P P, Grecu C, Ivanov A, et al. High-throughput switch-based interconnect for future SoCs[C]//Proceedings of 3rd IEEE International Workshop on System-on-Chip for Real-Time Applications. Washington, DC: IEEE CS, 2003: 304-310

[10] Scott S L, Thorson G M. The cray T3E network: adaptive routing in a high performance 3D torus[C]//Proceedings of Hot Interconnects IV. Stanford, CA: Stanford University, 1996: 147-156

[11] Dally J, Seitz C L. Deadlock-free message routing in multiprocessor interconnection networks[J]. IEEE Transactions on Computers, 1987, 36(5): 547-553

[12] 尹宝林,何自强,许光汉,等.离散数学[M].北京:高等教育出版社, 1998
Yin Baolin, He Ziqiang, Xu Guanghan, et al. Discrete mathematics[M]. Beijing: Higer Education Press, 1998(in Chinese)

[13] Jerraya A A, Tenhunen H, Wolf W. Guest editors' introduction: multiprocessor systems-on-chips[J]. IEEE Computer, 2005, 38(7): 36-40

[14] Chien A A. A cost and speed model for k-ary n-cube wormhole routers[J]. IEEE Transactions on Parallel and Distributed Systems, 1998, 9(2): 150-162

(责任编辑:赵海容)