k-ary n-cube 中的移动气泡流控策略*

王永庆,张民选
(国防科技大学 计算机学院,湖南 长沙 410073)

摘 要:在k-ary n-cube网络中,气泡流控是一种有效、实用的死锁避免技术,它不必依赖虚通道就能避免环网中出现的死锁问题。如果流控策略能感知到维度内缓冲区的总体使用情况,就能够更加高效地进行调度,从而提高网络性能。为了避免关键气泡机制引起的阻塞,提出了伪报文协议;结合伪报文协议,设计了移动气泡流控策略,它有效实现了维度内的全局资源感知能力。与局部气泡流控相比,路由器每条输入通道仅设置一个报文缓冲区就可以避免环网中的死锁,即最小资源需求减少了一半。网络模拟结果表明,该机制不会出现永久阻塞;在distribute、hotregion和uniform传输模式中,该机制可以有效提高网络吞吐率20%以上,并且在网络饱和后吞吐率依然维持稳定。

关键词:气泡流控;k-ary n-cube;互联网络;死锁;虚跨步
中图分类号:TP302 **文献标志码**:A **文章编号**:1001-2486(2012)06-0034-05

Moveable bubble flow control in k-ary n-cube

WANG Yongqing, *ZHANG Minxuan*
(College of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract: Bubble flow control is an efficient technique to avoid deadlock in torus networks without using virtual channels. If a flow control mechanism has knowledge of buffer utilization within a dimension, it can make resource allocation decisions based on global network conditions to improve network performance. The previous critical bubble scheme has a risk of blocking. To resolve this problem, a false packet protocol was presented, and a non-blocking moveable bubble scheme was designed, which is an improvement of critical bubble scheme with a requirement of one packet buffer at least, which halves the buffer requirement of two. Network simulation results show that this scheme is apparently better than the existing methods, avoids permanent blocking, displays a throughput improvement of more than 20% under distribute, hotregion and uniform traffic patterns, and maintains a steady throughput after network saturation without sharp drop.

Key words: bubble flow control; k-ary n-cube; interconnection networks; deadlock; virtual cut-through

当前计算机领域广泛采用并行处理方式,并行处理需要互联网络提供高效的通信支持。k-ary n-cube网络是一种备受关注的网络拓扑,它具有拓扑结构规整、结点度低和易实现等优点。不仅大量的片上网络采用二维网格拓扑结构,在当前的商用超级计算机中,k-ary n-cube网络也是主流的拓扑结构,比如Cray XT系列[1](3D torus),IBM Blue Gene/Q[2](5D torus)和Fujitsu K计算机[3](6D torus)。

在k-ary n-cube中如何解决死锁是网络研究和设计的一个关键课题。流控策略和路由算法应该尽可能提高系统的资源使用效率,尤其是要避免死锁的出现。

一个有意义的思想是采用流控技术来避免死锁。网络的流控策略用来判断路由函数选择的候选通道是否可用。按照使用的信息来源,流控机制可以分为全局感知和局部感知两种类型。局部感知网络流控仅依靠路由器节点局部信息分配网络资源,如根据本节点或者相邻节点的通道缓冲区空闲数量。全局感知网络流控则基于全局网络条件进行决策,如既使用本地也使用远程节点的通道缓冲区空闲数量。具有全局感知能力的流控机制可以优化网络使用效率,包括减少网络拥塞,提高性能,如延迟和吞吐率。但是这些机制通常需要一个万能的全局控制器,它需要随时收集和发布所有节点的信息,导致设计十分复杂;如果使用简化的局部实现,则可能会导致性能降低。

1 相关研究

在k-ary n-cube网络中,死锁不仅会出现在维度之间,也可能发生在一维内部的环上。本文关注的是后者。

* **收稿日期**:2012-07-11
基金项目:国家"863"高技术研究发展计划基金项目(2012AA01A301)
作者简介:王永庆(1973—),男,山东潍坊人,副研究员,博士,E-mail:yqwang@nudt.edu.cn

解决维内死锁的一种典型方法是使用分界线(dateline)技术[4],其缺点是要使用两条虚通道。

Carrión[5]提出了一种通过流控函数来避免死锁的机制,报文对下一级的缓冲需求由报文路由方向决定。Puente[6]明确提出了气泡流控(Bubble Flow Control,BFC)的思想,消除了 torus 网络中由于环绕导致的死锁。作者在采用虚跨步(virtual cut-through)交换实现自适应路由器时,仍然采用局部流控,路由器每个输入通道至少需要两个报文缓冲区。超级计算机 BlueGene/L[7]采用气泡流控方式实现逃逸通道。

实现全局气泡流控的难度在于需要一个复杂全局控制器,一方面它要随时收集有关网络的全局信息,另一方面还要实现资源竞争访问中的全局仲裁。在网络规模较大的情况下,这两个任务都难以实现。

CBS(Critical Bubble Scheme)机制[8]可以有效地实现全局 BFC,路由器输入缓冲区最少只需要一个输入报文缓冲空间就可以避免死锁。其基本思想是把报文缓冲区分为关键气泡(一个特殊的空闲缓冲区)和普通缓冲区,某个关键气泡仅在一个特定维度的单向环中流动。当报文下一步与当前方向属于同一维时,可以使用任意类型的空闲缓冲区;当报文行进维度发生变化时,则需要一个普通的空闲报文缓冲区,从而取得与使用全局控制器一样的效果。减少缓冲区需求,从而降低功耗,对于片上网络结构具有更加实用的意义。

如图 1 所示,在 4-ary 2-cube 网络中,通过采用 CBS 流控避免环内死锁,采用维序路由(Dimension-Order Routing,DOR)避免维间死锁,考虑输入通道仅有一个报文缓冲区的情况,图中黑色圆形阴影表示 x + 方向上关键气泡所在的位置。假定 P21 要向 P22 发送报文,首先要经过 R22 的 w 输入,如果采用的是局部流控,则需要 R21 或 R22 的 w 输入通道至少有两个空闲报文缓冲区,在本网络中无法进行通信。如果采用 CBS 机制,只要 R22 的 w 输入缓冲区空闲,R21 就可以发送报文。

虽然 CBS 机制实现了全局气泡流控机制,但是其关键气泡的流动规则可能导致某些报文被永久阻塞,这种风险并不是通常意义上资源相关导致的死锁,而是流控函数本身引起的。同样如图 1 所示,假定起初网络上没有数据流动。现在,P22 要给 P33 发送报文。按照 DOR 路由,P22 的报文首先要到达 R23 的 w 入口,由于 w 输入通道上的缓冲区只有一个关键气泡,没有其他空闲缓冲区,因此 P22 上的注入报文被流控函数阻止在注入队列中,等待关键气泡转移。但是,由于网络中 x + 方向上不存在其他从 R22 到 R23 的报文,关键气泡总是停留在 R23 的 w 通道上,导致 P22 的数据永远无法发出。

2　移动气泡流控

在上述例子中,没有资源相关导致的死锁,但是由于流控条件难以满足,导致报文被永久阻塞。这种阻塞不仅仅存在于只有一个输入报文缓冲区的情况。当输入缓冲区可以存放多个报文,而且维内使用多个关键气泡时,这些气泡可能会慢慢聚集到一个路由器的同一通道中,如果空闲缓冲区的数量等于关键气泡的数量,也会发生上述阻塞。

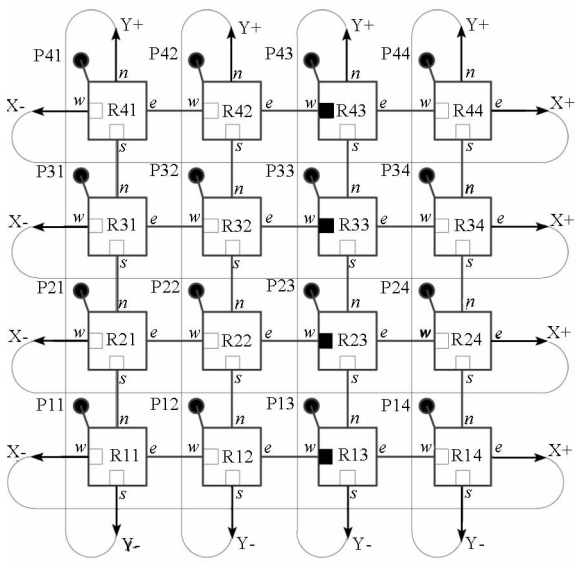


图 1　4 ×4torus 网络

Fig. 1　4 ×4 torus network

针对上述阻塞问题,我们提出了移动气泡流控(Moveable Bubble Scheme,MBS)机制,从而完善了关键气泡流控的思想,弥补了其流控引入的阻塞问题。MBS 的主要思想是避免关键气泡永久停留在一个固定的位置,导致某一输入通道没有普通空闲缓冲区。这不仅避免了永久阻塞,还提供了更好的公平性。

仍然以图 1 为例,当 x + 方向上没有 R22 到 R23 的报文流时,R23 上 w 入口的关键气泡不会移动,如果能够制造一个虚假的报文,让 R22 认为该报文从其 w 输入缓冲区流到了 R23 的 w 输入缓冲区,则可以让关键气泡转移到 R22 的 w 输入缓冲区,R23 中的关键气泡变成了普通缓冲区,这样,P22 就可以向 P33 发送报文。

实现 MBS 的关键,就是要避免关键气泡占据某输入通道的所有空闲缓冲区,并且不再转移。

要实现关键气泡的转移,就需要让上游路由器同方向上的输入缓冲区有空闲的普通缓冲空间时,把本地的关键气泡转移到上游路由器中,这样,在本地就会有一个空闲的普通缓冲区,来自任何维度的报文都可以使用该普通缓冲区来传输数据。

可见,转移本地输入通道中关键气泡需要上游路由器的帮助,把关键气泡转移到上游。如果上游路由器在该方向上有维内流动的报文(如同CBS假定的那样),自然会释放本输入通道中的关键气泡;如果上游没有这种报文,我们就要让上游路由器使用一个普通的空闲缓冲区来取代本地的关键气泡,实现关键气泡的转移。

2.1　伪报文协议

首先我们定义伪报文协议,它的主要作用就是产生一个临时报文,以便实现关键气泡的转移。

伪报文协议在链路直接连接的两个端口上执行,包括请求阶段和响应阶段。在请求阶段,一方发送请求报文,另外一方接收和处理请求报文;在响应阶段,接收到请求报文的一方发送响应报文,另外一方接收和处理响应报文。请求报文作为控制报文,不消耗信用,因而也不占用输入缓冲区,在接收端可以立即解释执行;而响应报文需要消耗信用,按照正常报文处理,但是仅在链路上两点之间传输,被接收后立即处理,不进行转发。为了描述的方便,我们参照图1中R21和R22来解释协议执行过程:

(1)R22通过w端口向R21发送请求报文;

(2)R21 e输入端口接收到请求报文后立即进行解析,如果此时R21有报文转发到e出口方向,则不再处理请求信息;如果此时R21没有报文转发到e出口方向,则自动生成一个响应报文发送到R22,并消耗e输出方向一个信用。

(3)R22在w入口上接收到响应报文并进行处理。

伪报文协议的主要目的是在没有有效数据报文流动时,产生一个临时的数据报文,协助关键气泡转移。该协议维持了路由器设计中通常使用的信用流控机制,不必使用特殊的信号线,从而可以应用于任意规模的网络中。

2.2　MBS机制的详细描述

假定k-ary n-cube中,两个相邻路由器使用两条方向相反的链路连接在一起,从而每一维中的链路构成了两个方向相反的单向环;假定采用信用流控机制,下游输入缓冲区的状态存放在上游的输出端口中,如信用数量和关键气泡数量,在多维网络中遵循DOR路由(实际上也可以采用其他机制,如转弯模型[9]等)。

(1)初始化:在每个单向环中,随机指定一个路由器(或者多个路由器)的一个空闲报文缓冲区为关键气泡,其他的都是普通缓冲区。

(2)流控规则:MBS定义了两条报文转发规则避免死锁和永久阻塞。①当报文继续在维内前进时,只要接收方至少有一个空闲报文缓冲区即可,无论是普通空闲缓冲区还是关键气泡;②注入新的报文,或者报文从一维转向另一维时,则要求接收方至少有一个普通的空闲缓冲区。如果目标缓冲区只剩有关键气泡,则不允许转发。

(3)关键气泡的转移:MBS机制的关键在于每个单向环中至少有一个空闲缓冲区,并且避免报文被关键气泡长久阻塞。为了叙述方便,我们仍然使用图1所示图形,但是假定输入通道上的空闲缓冲区数量为n,其中关键气泡数量为m,显然$m<=n$。按照报文的前进方向和目标通道输入缓冲区状态(假定目标通道为R32的s输入通道),分为4种情况:

①如果$m<n$,则目标缓冲区中至少还存在一个空闲的普通缓冲区,则允许来自R22中任何方向上的报文请求该通道。缓冲区中的关键气泡数量不变。

②如果$m=n$(且不为0),且R22的s输入端有报文流向R32。此时所有空闲缓冲区都是关键气泡,因而只允许不改变维度方向的报文请求使用该缓冲区,即只允许当前位于R22中s输入端口的报文来使用关键气泡。当R22中的报文流向R32的s输入端口后,R32中的关键气泡数量减1,而R22中的关键气泡数量加1,即实现了一个关键气泡的转移。

③如果$m=n$(且不为0),R22的s输入端有报文但是报文不流向R32。当R22 s输入缓冲区中的报文流出后,R32中的关键气泡数量减1,而R22中的关键气泡数量加1,即实现了关键气泡的转移。当R22返回信用给R12时,会通知其本地缓冲区中关键气泡数量的变化情况。这样,R32的s输入通道上$m<n$,有了一个空闲的普通缓冲区。

④如果$m=n$(且不为0),R22的s输入端没有报文(但是有空闲的普通缓冲区)。在每个路由器输出端(如R22的n输出端口)设置一个计时器,监视下游输入端上(如R32的s输入端口)m和n的数值。当$m\neq n$时,计时器禁止;当$m=n$时,计时器启动。当计时器出现超时,则执行前面所述的伪报文协议。本例中,即在R22的s端

口和 R12 的 n 端口之间执行该协议。当 R22 s 输入端接收到响应报文后,释放的缓冲区会被标记为关键气泡,并在信用返回时通知 R12 的 n 输出端;同时 R22 通知其 n 输出端减少下游关键气泡的数量,亦会使得 $m < n$ 成立。

在 CBS 机制中,如果出现上面的③和④,则无法进行关键气泡的转移;而在 MBS 机制中,则仍然可以保证关键气泡的移动,从而避免其长时间停留在一个地方。

在 MBS 机制中,仅在④中执行伪报文协议。请求阶段可以使用一个最小的报文,也可以在常规报文中设置一个特定位标记来携带请求,如果采用后者,则请求阶段不会消耗额外的通信带宽(位标记相对于报文长度来说很小);在响应阶段,只有在没有正常数据流动时才会发出响应报文,如果有正常数据流动,则不必额外发送响应报文,因此响应阶段不会影响正常数据传输。也就是说,伪报文协议的执行不会使 MBS 比 CBS 有明显的通信开销。

3　路由器结构需求

在典型的虚跨步路由器中,到达的报文首先存储在输入缓冲区中,然后以 FIFO 方式进行处理,路由模块负责计算报文的输出端口,当报文到达 FIFO 首部,仲裁单元配置交叉开关,设置报文的输出路径,然后报文就经过交叉开关流到输出端。

为了支持 MBS,需要路由器结构进行下面的设置:

(1)每个输出端口设置信用计数器 C,表示下游路由器输入端口中空闲缓冲区的数量。当接收到下游发来的信用返回信号时,C 增加;当向下游发送报文时,C 减少。实际上这也是当前信用机制常用的实现方法。

(2)每个输出端口设置计数器 B,表示下游路由器输入端口中关键气泡的数量,当接收到下游发来的通知信号时,B 增加;当下游剩余信用数量 C 等于 B 时,如果有报文发出,则在信用减少的同时 B 也减少;当执行伪报文协议时,接收到响应报文的输入端口会通知其同方向的输出端口,减少 B 的数量。B 不会大于下游路由器可用信用 C。

(3)释放的信用返回到上游发送端口时需要携带新增关键气泡的数量,即该返回信用是否属于关键气泡。

(4)在输出端口增加超时计数器 T。当 $C > B$ 时,超时计数器清 0;当 $C = B$ 时,超时计数器开始计时;当计数器到达某一阈值时产生超时信号,超时信号会传递给所在路由器同维度反向的输出端口,如图 2 所示。

(5)发送端口添加请求报文和响应报文产生逻辑。端口在接收到 T 超时信号后,发送请求报文,不需要消耗信用;在端口接收到请求报文后,发送响应报文,要消耗一个信用。

(6)接收端口添加请求报文和响应报文接收逻辑。在接收到请求报文时,产生请求信号,触发响应报文;在接收到响应报文时,丢弃报文内容,释放其信用,返回该信用时携带一个关键气泡标记,同时发送 B 减小信号到所在路由器同维度同向的输出端口,如图 2 所示。

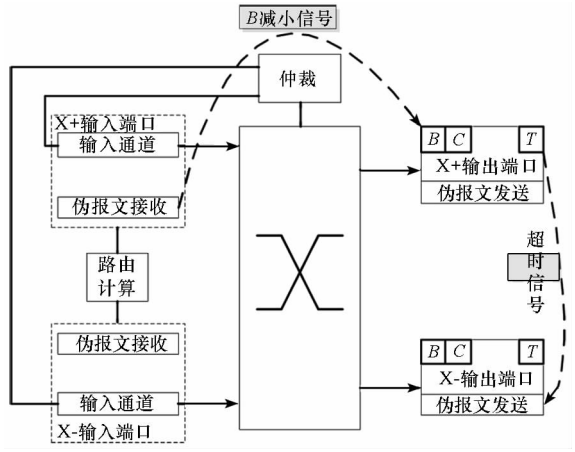


图 2　路由器简单示意图

Fig. 2　A simple router supporting MBS

4　模拟结果

为了评价 MBS 机制,我们采用互联网络模拟器 INSEE[10] 比较了几种不同的气泡流控策略。INSEE 采用的是轻量级设计架构,可用来方便地研究互联网络性能。其内部已经实现了几种路由器架构,包括使用气泡流控和基于虚通道的路由器。其中,气泡流控采用的是局部信息模式,我们称之为 BLOC。然后我们实现了 MBS 和 CBS 流控,采用了固定长度为 16 Phit 的报文。Phit 是在物理链路上并行传输的位宽。路由器输入队列长度分别设计为 1 个和 2 个报文空间,分别是全局流控和局部流控所需的最小缓冲区空间。

在网络拓扑上,我们选择了 8 × 8 torus,链路采用双向物理通道。每条物理通道只有一条虚通道,每个路由器有一个注入端口。在 torus 网路中,气泡流控只需要一条虚通道和 DOR 路由就能够保证网络的无死锁特性。

当多个输入通道请求同一输出端口时,使用 round-robin 仲裁策略。为了消除模拟的偶然性,所有模拟均运行了 15 次后取平均值,并设置其中

warm-up 时间为 25000 周期。为了观察网络到达饱和后的流控效果,注入负载速率变化范围为 0 ~ 1 flit/cycle/node。我们选择了几种不同的传输模式,分别是: uniform, distribute, transpose 和 hotregion。

模拟 MBS 时分别选择输入缓冲区为 1 个报文空间(MBS1)和两个报文空间(MBS2)。BLOC 只能使用两个输入缓冲区运行。由于存在阻塞,CBS 使用一个输入报文缓冲区时难以完成模拟,因此这种情况被忽略,而在使用两个输入报文缓冲区时可以完成运行(CBS2)。

图 3 分别是在 4 种不同传输模式下的吞吐率随着输入负载的变化情况。

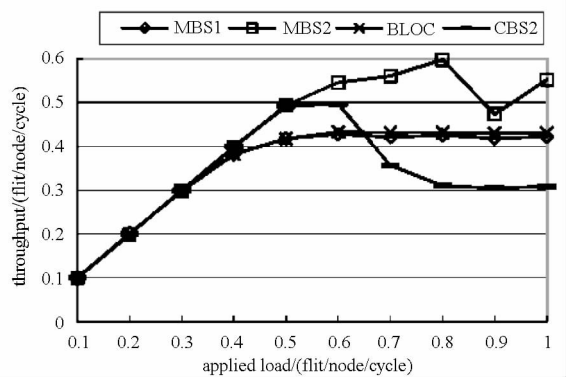
从图中可以看出,几乎在所有情况下 MBS2 都能够比其他流控取得更高的性能。在 distribute 传输模式下,达到的峰值吞吐率接近 0.55。因为在这种负载模式下,资源使用比较均衡,相比其他负载网络能够达到更高的吞吐率。无论在 uniform, distribute,还是 hotregion 模式下,MBS2 性能的提高都超过其他方法 20% 以上,这充分展示了其全局感知的优势,因为只需要目标缓冲区有一个普通的报文缓冲区就允许报文进行转发,而在局部气泡流控中却需要等待有两个空闲报文缓冲区,从而提高了网络使用效率。

在 transpose 模式下,所有使用两个输入缓冲区的流控方式都具有类似的效果,并且在此模式下吞吐率都明显低于其他传输模式,因为在该模式下,网络的资源使用很不均衡,在某些路径上通信压力太大,导致整体吞吐率降低。

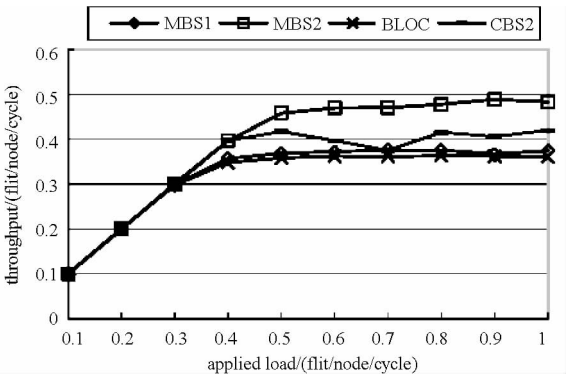
MBS1 和 BLOC 的曲线都比较接近。BLOC 在 uniform, distribute 和 transpose 模式下性能略高于 MBS1,毕竟它使用了两个报文的输入缓冲区。但是在 hotregion 模式下,所有使用全局气泡流控的机制都能够取得比 BLOC 更高的吞吐率,尽管 MBS1 只使用了一个报文空间。这一点充分显示了全局流控的优势。

MBS2, MBS1 以及 BLOC 在达到峰值后通常能够维持在该水平附近,即使注入的负载已经使网络饱和,例外的情况发生在 transpose 模式下,此时所有的曲线在达到某个峰值后都有一点下降的趋势。当注入队列满的时候,报文的产生就会停止,就是说,达到饱和后,实际产生的负载速度已经小于图中所示的施加负载。

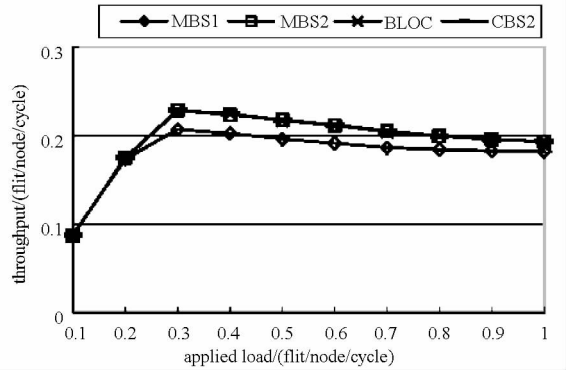
CBS 流控具有的一个明显特点是,除了 hotregion 模式,当吞吐率达到饱和点后都会有明显的下降。虽然它也是一种全局气泡流控,但是其性能的变化很大,超过饱和后,其性能甚至比不上使用局部流控的 BLOC。正是我们前面提到的阻塞问题,导致其性能的不稳定。



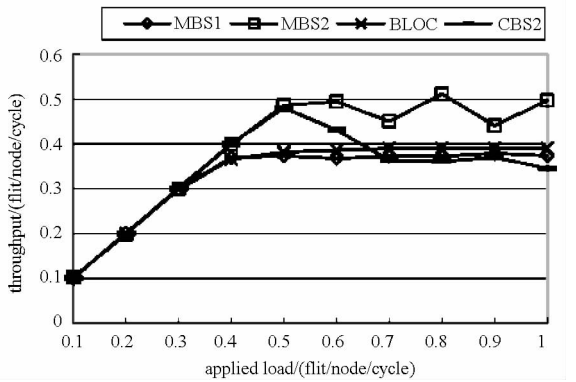
(a) Distribute 传输模式



(b) Hotregion 传输模式



(c) Transpose 传输模式



(d) Uniform 传输模式

图 3 各传输模式下吞吐率随负载变化情况

Fig. 3 Throughput under different traffic patterns

(下转第 53 页)

参考文献(References)

[1] Suresh D, Najjar W, Vahid F, et al. Profiling tools for hardware/software partitioning of embedded applications [C]// Proceedings of the ACM SIGPLAN Conference on Language, Compiler, and Tool for Embedded Systems, San Diego, USA. 2003: 189 – 198.

[2] Cardoso P, Diniz C, Weinhardt M. Compiling for reconfigurable computing: a survey [J]. ACM Computing Surveys, 2010, 42(4): 1 – 65.

[3] Yoon J, Shrivastava A, Park S, et al. SPKM: A novel graph drawing based algorithm for application mapping onto coarse-grained reconfigurable architectures [C]//Proceedings of the Asia South Pacific Design Automation Conference, Seoul, Korea: 2008:776 – 782.

[4] 王大伟,窦勇,李思昆. 核心循环到粗粒度可重构体系结构的流水化映射[J]. 计算机学报, 2009, 32(6): 1089 – 1099.
WANG Dawei, DOU Yong, LI Sikun. Loop kernel pipelining mapping onto coarse-grained reconfigurable architecture for data-intensive applications [J]. Journal of Computers, 2009, 32(6):1089 – 1099. (in Chinese)

[5] Joao M, Cardoso P. Dynamic Loop pipelining in data-driven architectures [C]//Proceedings of the 2nd Conference on Computing Frontiers, New York, NY, USA,2005:106 – 115.

[6] Kim Y, Mahapatra R. Hierarchical reconfigurable computing arrays for efficient CGRA-based embedded systems [C]// Proceedings of the 46th Annual Design Automation Conference, San Francisco, USA,2009:826 – 831.

[7] Balasa F, Zhu H, Luican I. Computation of storage requirements for multi-dimensional signal processing applications [J]. IEEE Transactions on Very Large Scale Integration Systems, 2007, 15(4): 447 – 460.

[8] LooPo - Loop parallelization in the polytope model [EB/OL]. University of Passau, 2012. [2012 – 5 – 2] http://www.fmi.uni-passau.de/loopo.

[9] ZHAO P, LI S K, WANG D W, et al. A new application feature analysis approach for system-on-chip hardware/software partitioning[C]//Proceedings of the International Congress on Image and Signal Processing, Sanya, China. 2008: 630 – 634.

[10] 窦勇,邬贵明,徐进辉,等. 支持循环自流水的粗粒度可重构阵列体系结构[J]. 中国科学, 2008. 38(4): 579 – 591
DOU Yong, WU Guiming, XU Jinhui, et al. A coarse-grained reconfigurable computing architecture with loop self-pipelining [J]. Science in China. 2008, 38(4): 579 – 591. (in Chinese)

[11] Bastoul C. Code generation in the polyhedral model is easier than you think [C]//Proceedings of the IEEE International Conference on Parallel Architecture and Compilation Techniques, Washington, DC, USA,2004: 7 – 16.

[12] 赵鹏, 王大伟, 李思昆. 面向 SoC 任务分配的应用程序存储需求量分析方法[J]. 电子学报, 2010. 38(3): 541 – 545.
ZHAO Peng, WANG Dawei, LI Sikun. Research on memory size estimation of application programs for system-on-chip task allocation[J]. Journal of Electronics, 2010, 38(3):541 – 545. (in Chinese)

(上接第 38 页)

5　结论

在互连网络中,全局气泡流控比局部流控有着潜在的优势。局部 BFC 需要接收缓冲区至少有两个报文的空间,而全局 BFC 只要有一个空闲报文空间即可避免死锁。CBS 机制首先提出了一种有意义的全局 BFC 实现方式,但是同时有阻塞的风险。在本文中,我们首先提出了伪报文协议来处理某些路径上没有报文流动的状况,然后结合该协议设计了移动气泡流控机制。通过使气泡不断移动,避免了阻塞,有效实现了全局气泡流控。实验结果表明,我们所提的机制在只有一个报文空间时网络是无阻塞的,而在使用两个报文缓冲空间的情况下,其吞吐率明显高于 BLOC 和 CBS 流控方式,除了 Transpose 传输模式外,MBS2 提高的幅度均大于 20% 。

参考文献(References)

[1] Alverson R, Roweth D, Kaplan L. The Gemini system interconnect [C]//Proceedings of the 2010 18th IEEE Symposium on High Performance Interconnects. Washington, DC: IEEE Computer Society, 2010:83 – 87.

[2] Chen D, Eisley N A, Heidelberger P, et al. The IBM blue gene/q interconnection fabric[J]. IEEE Micro, 2012, 32:32 – 43.

[3] Ajima Y, Sumimoto S, Shimizu T. Tofu: a 6d mesh/torus interconnect for exascale computers [J]. Computer, 2009, 42:36 – 40.

[4] Dally W, Towles B. Principles and practices of interconnection networks [M]. San Francisco: Morgan Kaufmann Publish, 2003.

[5] Carrión C, Beivide R, Gregorio J A, et al. A flow control mechanism to avoid message deadlock in k-ary n-cube networks [C]// Proceedings of the Fourth International Conference on High-Performance Computing. Washington, DC: IEEE Computer Society, 1997: 322 – 329.

[6] Puente V, Izu C, Beivide R, et al. The adaptive bubble router [J]. Journal of Parallel and Distributed Computing, 2001, 61: 1180 – 1208.

[7] Adiga N R, Blumrich M A, Chen D, et al. Blue gene/l torus interconnection network [J]. IBM Journal Res. Dev., 2005, 49: 265 – 276.

[8] Chen L, Wang R, Pinkston T M. Critical bubble scheme: an efficient implementation of globally aware network flow control [C]// Proceedings of the 2011 IEEE International Parallel & Distributed Processing Symposium, Washington, DC: IEEE Computer Society, 2011:592 – 603.

[9] Glass C J, Ni L M. The turn model for adaptive routing [J]. Journal of ACM, 1994, 41(5):874 – 902.

[10] Navaridas J, Alonso J M, Pascual J A, et al. Simulating and evaluating interconnection networks with insee [J]. Simulation Modelling Practice and Theory, 2011, 19(1):494 – 515.