基于闭合最小图划分模型的多作业分配优化方法

张拥军 林宇斐
(国防科学技术大学计算机学院 长沙 410073)

摘 要 随着并行计算系统规模的增大和复杂度的提高,已有的多作业分配方式可能导致较长的通信延迟和严重的通信竞争。针对这一问题,提出了一种基于闭合最小图划分模型的多作业分配优化方法。该方法以最小化通信延迟和消除通信竞争为出发点,通过建立闭合最小图划分模型,将多作业分配优化问题转化成闭合最小图划分问题,并设计闭合最小图划分算法来获得优化的多作业分配方案。

关键词 多作业分配,图划分,通信竞争,网络直径
中图法分类号 TP302 **文献标识码** A

Multi-job Assignment Optimization Approach Based on Closed Minimum Graph-partitioning Model

ZHANG Yong-jun LIN Yu-fei
(College of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract Due to the augment of the parallel computing system size and the increase of its complexity, the existing multi-job assignment approaches can cause severe communication latency and contention. In order to solve this problem, a new multi-job assignment approach based on a closed minimum graph-partitioning model was proposed. To minimize the communication latency and eliminate the communication contention, this approach translates the multi-job assignment optimization problem to the closed minimum graph-partitioning problem by building a closed minimum graph-partitioning model, and designs the closed minimum graph-partitioning algorithm to obtain an optimized multi-job assignment scheme.

Keywords Multi-job assignment, Graph-partitioning, Communication contention, Network diameter

为了适应自然灾害预测、气候变化分析和外层空间探索等大规模并行应用日益增长的性能需求,并行计算系统的规模正在不断扩大。当前世界排名第一的超级计算机"天河二号"已经拥有2120000个计算核心[1]。有专家预测,在2018年,超级计算机的计算核心数将会达到10^8[2]。

并行计算系统拥有丰富的计算资源,其上往往同时运行多个不同的作业(系统的用户和维护者通常把系统上运行的并行程序称为作业)。合理地为多个作业分配计算资源以满足作业的性能需求,对于用户来说十分重要[3]。然而,目前已有的块分配、循环分配和循环块分配等分配方式,可能导致较长的通信延迟和严重的通信竞争[4];并且,随着系统规模的扩大和系统复杂度的提升,这些通信问题将愈发严重[5]。

在此背景下,本文提出了一种新的多作业分配优化方法:基于闭合最小图划分模型的多作业分配优化方法。本文首先形式化地描述了多作业分配;然后提出了多作业分配优化的目标;接下来建立闭合最小图划分模型,将复杂抽象的多作业分配优化问题转化为闭合最小图划分问题;最后提出了闭合最小图划分算法用于求解闭合最小图划分问题。基于"天河-1A"子系统的实验表明,对于NPB测试集[6],本文提出的多作业分配优化方法具有很好的性能提升效果。

1 多作业分配

一个大规模并行系统上往往同时运行多个作业,而一个作业往往需要由多个任务并行执行完成[7]。本文称系统上运行的所有作业构成的集合为作业集。为了具体表示某一个作业,本文将作业集表示成序列,记为$P=\langle P_1,P_2,\cdots,P_n\rangle$。而针对一个作业通常包含多个任务的现象,本文定义了作业集的属性的概念。

定义1(作业集的属性) 对于一个作业集$P=\langle P_1,P_2,\cdots,P_n\rangle$,作业$P_i$中的任务个数为$num_i$,那么序列$\langle num_1,num_2,\cdots,num_n\rangle$称为该作业集的属性。

需要说明的是,并行计算系统由物理结点(包括计算结点和路由结点)以及通信链路组成,每个计算结点包含一个或多个处理单元(核)。计算结点一般不和其它计算结点直接连接,源计算结点发出的消息需要经过路由结点和通信链路才能到达目标计算结点[12]。接下来,我们给出多作业分配的定义。

定义2(多作业分配) 将属性为$\langle num_1,num_2,\cdots,num_n\rangle$的作业集$P=\langle P_1,P_2,\cdots,P_n\rangle$中的所有任务分配到并行系统S的各个处理单元上的过程称为多作业分配。

到稿日期:2013-08-14 返修日期:2013-11-04 本文受国家自然科学基金(60921062)资助。
张拥军(1972—),男,博士,副研究员,主要研究方向为高性能计算,E-mail:yjzhang@nudt.edu.cn;林宇斐(1985—),女,博士,主要研究方向为高性能计算。

多作业分配优化问题就是要找到一种多作业分配方案，使得每个作业的执行时间尽量短。不失一般性，我们假设某个处理单元上分配的任务数不超过 1。也就是说，本文在针对多作业分配进行优化时并不考虑作业间的负载平衡问题，而是研究如何进行分配，才使得对于每个作业来说，其通信导致的额外开销尽量小，例如，如何使得一个作业内的任务之间的通信延迟尽量小，如何消除各个作业在通信时的竞争等等。

2 多作业分配优化的目标

定义 3(多作业分配) 记系统 S 中分配给属性为$\langle num_1, num_2, \cdots, num_n\rangle$的作业集 $P=\langle P_1, P_2, \cdots, P_n\rangle$的计算结点集合分别为 $N_1, N_2, \cdots, N_n$，未被分配的剩余计算结点集合为 N_{n+1}。$\langle N_1, N_2, \cdots, N_{n+1}\rangle$称为针对作业集 P 的多作业分配，若其满足：

- $N_1 \cup N_2 \cup \cdots \cup N_n \cup N_{n+1} = N$，$N$ 为系统 S 中计算结点的集合；
- $\forall i,j,i,j=1,2,\cdots,n+1, i\neq j, N_i \cap N_j = \emptyset$；
- $\forall P_i, i=1,2,\cdots,n, num_i = |N_i|$。

值得注意的是，定义 3 所说的计算结点只包含一个单核处理器。对于更一般的情况(即一个计算结点包含多个共享内存的多核处理器)，假设一个计算结点包含的核(处理单元)的数目为 num_{core}。此时，只需将这样的计算结点视为一个只包含一个单核处理器的计算结点，然后求解针对属性为$\langle \frac{num_1}{num_{core}}, \frac{num_2}{num_{core}}, \cdots, \frac{num_n}{num_{core}}\rangle$的作业集的多作业分配问题即可。因此，在接下来的讨论中，我们假设一个计算结点只包含一个单核处理器，并不再对上述更一般的情况进行说明。

作业的执行时间是由其各个任务的计算开销与通信开销决定的，第 1 节已经假设了各个处理单元上分配的任务数不超过 1，因此，本文并不考虑通过作业分配对计算开销进行优化，例如负载平衡问题等，而是聚焦于通信开销的优化。通信带宽、通信延迟和通信时发生的网络资源竞争是影响通信开销的 3 个重要因素。作业在并行系统的计算结点上的分配情况，一方面决定了作业之间是否存在网络资源竞争，另一方面决定了各个计算结点之间的通信延迟。因此，我们希望通过优化作业的分配，消除作业之间的网络竞争，并最小化作业内部的通信延迟。网络直径是两个结点间最短路径所经过的链路数或跳步数的最大值[10]，很多文献[8,9]常用它来衡量通信延迟。基于上述讨论，本文给出多作业分配优化问题的优化目标：

对于系统 S 和作业集 P，找到一种多作业分配$\langle N_1, N_2, \cdots, N_{n+1}\rangle$，使得在该种分配方式下，

- 作业集 $P=\langle P_1, P_2, \cdots, P_n\rangle$中各个作业之间不存在网络资源竞争；
- $\forall i, i=1,2,\cdots,n$，作业 P_i 所分配的计算结点间的网络直径最小化。

3 闭合最小图划分模型

我们将系统以图的方式进行描述，通过一系列概念和定理，建立闭合最小图划分模型。该模型将抽象的多作业分配优化问题转化为具体的闭合最小图划分问题。

3.1 图划分

本节将给出系统的拓扑图的概念，并基于拓扑图描述图划分以及与多作业分配对应的图划分的概念，从而将多作业分配问题等价为图划分问题。

定义 4(拓扑图) 对于一个并行计算系统 S，定义 $G=(V,E)$为其拓扑图，其中：

- V 是结点集合，结点 $v\in V$ 代表系统 S 中的一个物理结点；
- E 是边的集合，边$\langle (u,v) \mid u,v\in V\rangle \in E$ 表示 u,v 所对应的物理结点有通信链路相连。

由于系统 S 中的结点包括计算结点和路由结点，因此本文记 V^C 和 V^R 分别为图 G 中系统 S 的计算结点和路由结点对应的结点构成的集合。显然，$V=V^C \cup V^R$。

定义 5(图划分) 对于图 G 和 n' 个图 $G_i=(V_i,E_i)$，$i=1,2,\cdots,n'$，如果它们满足：

- $V_1 \cup V_2 \cup \cdots \cup V_{n'} = V$；
- $\forall i,j,i\neq j$，有 $V_i \cap V_j = \emptyset$。

则称$\langle G_1, G_2, \cdots, G_{n'}\rangle$为图 G 的图划分，图 G_i 为图 G 的子图。

与系统的拓扑图类似，记 V_i^C 和 V_i^R 是子图 G_i 中对应的计算结点和路由结点的集合，显然 $V_i=V_i^C \cup V_i^R$，$i=1,2,\cdots,n'$。

定义 6 对于系统 S 和其上针对作业集 $P=\langle P_1, P_2, \cdots, P_n\rangle$的多作业分配$\langle N_1, N_2, \cdots, N_{n+1}\rangle$以及系统 S 的拓扑图 G 的图划分$\langle G_1, G_2, \cdots, G_{n'}\rangle$，如果它们满足：

- $n'=n+1$；
- 对于子图 G_i，$i=1,2,\cdots,n$，N_i(作业 P_i 分配的计算结点)$\subseteq V_i^C$ 对应的计算结点。

则称图划分$\langle G_1, G_2, \cdots, G_{n+1}\rangle$是与多作业分配$\langle N_1, N_2, \cdots, N_{n+1}\rangle$对应的图划分，且子图 $G_i(i=1,2,\cdots,n)$，为有效子图，子图 G_{n+1} 为剩余子图。

此时，多作业分配问题等价于图划分问题。

3.2 闭合子图

对于一个属性为$\langle 4,4,4,4\rangle$的作业集，图 1 是一种该作业集在 4×4 的 2D-mesh 上的分配方案。网络资源竞争的原因是两条不同的消息试图同时使用同一个路由器的同一个输入(或输出)通道[12]。以维序路由算法[10]为例，不难发现，在图 1 这种分配方案下，该作业集中的各个作业内部进行通信时不会经过相同的路由器。因此，这 4 个作业之间不会发生网络资源的竞争。

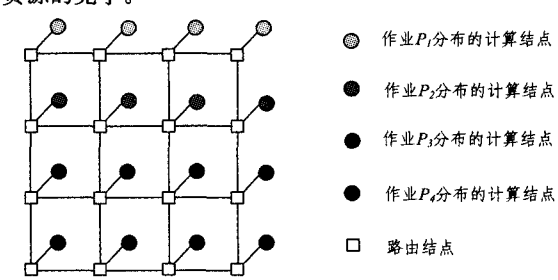

图 1 属性为$\langle 4,4,4,4\rangle$在 4×4 的 2D-mesh 的分配

记 V_S^C 为系统 S 内所有计算结点构成的集合，V_S^R 为系统 S 内所有路由结点构成的集合。为了便于表示，我们为每个计算结点集合定义如下函数，用于计算在路由规则 rule 下其内任意计算结点之间通信所经过的路由结点的集合：

$$f_{rule}: 2^{V_S^C} \rightarrow 2^{V_S^R}$$

该函数的自变量是系统 S 的计算结点构成的集合 s，函

数的值是在路由规则 rule 下 s 内任意计算结点之间通信经过的路由结点的集合。对于系统 S 和其上针对作业集 P 的多作业分配$\langle N_1,N_2,\cdots,N_{n+1}\rangle$，如果 $\forall P_i,P_j,i,j=1,2,\cdots,n,i\neq j$，有 $f_{rule}(N_i)\cap f_{rule}(N_j)=\emptyset$，那么就说明在该多作业分配方案下，作业集 P 的各个作业在通信时不会经过相同的路由结点，从而不存在网络资源竞争，即满足了多作业分配优化的第一个目标。

下面，我们首先提出闭合子图的概念，然后给出定理，证明了：如果与多作业分配$\langle N_1,N_2,\cdots,N_{n+1}\rangle$对应的图划分$\langle G_1,G_2,\cdots,G_{n+1}\rangle$中的有效子图都是闭合子图，那么该多作业分配下的各个作业之间是不存在网络资源竞争的。

定义 7(闭合子图)　对于系统拓扑图 G 和其子图 $G_i,i=1,2,\cdots,n'$，如果 V_i^C 对应的任意计算结点之间互相通信时所经过的路由结点在 G_i 中对应的结点仍在该子图内，即 V_i^C 对应的任意计算结点之间互相通信时所经过的路由结点在 G_i 中对应的结点$\in V_i^R$，那么 G_i 是一个闭合子图。

定理 1　如果与多作业分配$\langle N_1,N_2,\cdots,N_{n+1}\rangle$对应的图划分$\langle G_1,G_2,\cdots,G_{n+1}\rangle$中的有效子图都是闭合子图，那么在该多作业分配下各个作业之间不存在网络资源竞争。

证明：根据定义 5，对于图划分$\langle G_1,G_2,\cdots,G_{n+1}\rangle$，有 $V_1\cap V_2\cap\cdots\cap V_{n+1}=\emptyset$。那么，显然有 $V_1^R\cap V_2^R\cap\cdots\cap V_{n+1}^R=\emptyset$。根据定义 7，如果有效子图 G_i 是闭合子图，那么 G_i 中 $V_i^C(i=1,2,\cdots,n)$对应的计算结点之间互相通信时所经过的路由结点在 G_i 中对应的结点$\in V_i^R$。因此，对于与图划分$\langle G_1,G_2,\cdots,G_{n+1}\rangle$对应的多作业分配$\langle N_1,N_2,\cdots,N_{n+1}\rangle$，有 $f_{rule}(N_1)\cap f_{rule}(N_2)\cap\cdots\cap f_{rule}(N_n)=\emptyset$，即各个作业通信时不会经过相同的路由结点。因此，在该多作业分配下各个作业之间不存在网络资源竞争。

定理 1 说明，如果图划分中的有效子图都是闭合子图，那么多作业分配的第一个优化目标得到满足。

3.3　最小子图

在本节中，我们将分析多作业分配的第二个优化目标。多作业分配的第二个优化目标是使得各个作业所分配的计算结点间的网络直径最小化，因此我们给出子图并行度的概念并基于此给出最小直径和最小子图的概念。

定义 8(子图并行度)　对于系统 S 的拓扑图 G 和其子图 $G_i,i=1,2,\cdots,n'$，子图 G_i 中对应于系统 S 的计算结点的结点个数为 $degree_i$，称 $degree_i$ 为子图 G_i 的并行度。

定义 9(最小直径)　在拓扑图 G 中所有并行度为 $degree_i$ 的子图的直径的最小值称为并行度 $degree_i$ 的最小直径。

定义 10(最小子图)　对于系统 S 的拓扑图 G 和其并行度为 $degree_i$ 的子图 $G_i,i=1,2,\cdots,n'$，若子图 G_i 的并行度 $degree_i>num_i$，且子图 G_i 的直径等于并行度 num_i 的最小直径，那么子图 G_i 被称为满足 num_i 并行度的最小子图。

不难发现，如果图划分中的任意有效子图 $G_i(i=1,2,\cdots,n)$都是满足 num_i 并行度的最小子图，那么在其对应的多作业分配下，作业 P_i 的 num_i 个任务所分配的计算结点间的网络直径最小，即多作业分配的第二个优化目标得到满足。

3.4　闭合最小图划分

基于第 3.2 节和第 3.3 节的分析，本节给出闭合最小子图的概念：

定义 11(闭合最小子图)　对于系统拓扑图 G 和其子图 $G_i,i=1,2,\cdots,n$，如果 G_i 是闭合子图且是满足 num_i 并行度的最小子图，那么 G_i 是一个闭合最小子图。

根据闭合最小子图的概念，不难定义闭合最小图划分：

定义 12(闭合最小图划分)　对于系统 S 和其上针对作业集 $P=\langle P_1,P_2,\cdots,P_n\rangle$的多作业分配$\langle N_1,N_2,\cdots,N_{n+1}\rangle$，图划分$\langle G_1,G_2,\cdots,G_{n+1}\rangle$是与该多作业分配对应的系统 S 的拓扑图 G 的图划分，如果 $\forall i,i=1,2,\cdots,n$，子图 G_i 是一个闭合最小子图，那么该图划分是一个闭合最小图划分。

一方面，如果图划分中的有效子图都是闭合子图，那么在其对应的多作业分配下，各个作业相互之间不存在竞争；另一方面，满足 num_i 并行度的最小子图就意味着该子图的任意 num_i 个结点对应的计算结点间的网络直径达到最小值。此时，多作业分配优化问题转化为寻找系统拓扑图的闭合最小图划分问题。

4　典型拓扑结构闭合最小图划分定理

本节将针对 2D-mesh 和树形结构对应的拓扑图，考虑确定性路由算法(2D-mesh 以维序路由算法为例)，研究它们的闭合最小图划分定理。为了描述的方便，我们用 2D-mesh 拓扑图以及树拓扑图指代采用 2D-mesh 以及树形网络的并行系统所对应的拓扑图。

4.1　2D-mesh

在 2D-mesh 网络中，每个计算结点连接着一个路由结点，路由结点之间则相互连接构成二维网格，同维相距最远的两个路由结点不环接[10]。同前面的描述一样，人们通常用 $k_1\times k_2$ 表示 2D-mesh 的规模，其中 k_1 和 k_2 表示 X 维和 Y 维上路由结点的个数。不难发现，图 1 就是 2D-mesh 拓扑图，图中圆圈表示的结点对应于计算结点，而方块表示的结点对应于路由结点。线性阵列是 2D-mesh 特例：当 $k_2=1$ 时，2D-mesh 成为线性阵列。

定理 2　对于 2D-mesh 拓扑图 G 及其子图 $G_i,i=1,2,\cdots,n'$，如果 G_i 是一个 2D-mesh 拓扑图，那么 G_i 是一个闭合子图。

证明：如果子图 $G_i(i=1,2,\cdots,n')$是 2D-mesh 拓扑图，根据维序路由的路由规则可知，对于 2D-mesh 中的任意两个计算结点之间按维序路由规则通信所经过的结点仍在这个 2D-mesh 内，因此对于子图 G_i，V_i^C 对应的计算结点之间互相通信时所经过的路由结点在 G_i 中对应的结点仍在该 2D-mesh 内。根据定义 7，子图 G_i 是一个闭合子图。

定理 3　对于 2D-mesh 拓扑图 G，并行度 num_i 的最小直径为$\lceil\sqrt{num_i}\rceil+\left\lceil\frac{num_i}{\lceil\sqrt{num_i}\rceil}\right\rceil-2$。

证明：当 $\sqrt{num_i}$ 为整数时，并行度 num_i 的最小直径为 $\sqrt{num_i}\times\sqrt{num_i}$ 的 2D-mesh 的直径，即 $2\sqrt{num_i}-2$。此时 $\lceil\sqrt{num_i}\rceil=\left\lceil\frac{num_i}{\lceil\sqrt{num_i}\rceil}\right\rceil$，故定理成立；

当 $\sqrt{num_i}$ 不为整数时，需要分情况讨论：

· 若$\lfloor\sqrt{num_i}\rfloor^2<num_i\leqslant\lfloor\sqrt{num_i}\rfloor\lceil\sqrt{num_i}\rceil$，并行度 num_i 的最小直径为$\lfloor\sqrt{num_i}\rfloor\times\lceil\sqrt{num_i}\rceil$的 2D-mesh 的直

径，即$\lfloor\sqrt{num_i}\rfloor+\lceil\sqrt{num_i}\rceil-2$。又由于此时$\left\lceil\frac{num_i}{\lceil\sqrt{num_i}\rceil}\right\rceil=\lfloor\sqrt{num_i}\rfloor$，故定理成立；

· 若$\lfloor\sqrt{num_i}\rfloor\lceil\sqrt{num_i}\rceil<num_i<\lfloor\sqrt{num_i}\rfloor^2$，并行度$num_i$的最小直径为$\lceil\sqrt{num_i}\rceil\times\lceil\sqrt{num_i}\rceil$的2D-mesh的直径，即$2\lceil\sqrt{num_i}\rceil-2$。又由于此时$\left\lceil\frac{num_i}{\lceil\sqrt{num_i}\rceil}\right\rceil=\lceil\sqrt{num_i}\rceil$，故定理成立。

4.2 树

在树形结构中，树的叶子结点是计算结点，中间结点是路由结点[10]。图2所示为一个4层二叉树，图中圆圈表示的结点对应于计算结点，而方块表示的结点对应于路由结点。在树形结构中，同一层的路由器之间没有直接相连的物理链路，同一层的路由器通信时必须通过上行链路经上层路由器转发，再通过下行链路到达同一层的其他路由器。对于图2所示的4层二叉树，上行链路和下行链路都是确定的，也就是说路由算法是确定性路由算法。

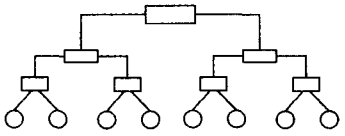

图2 树拓扑图

定理4 对于树拓扑图G及其子图$G_i(i=1,2,\cdots,n')$，如果G_i是一个树拓扑图，那么G_i是一个闭合子图。

证明：已知子图$G_i(i=1,2,\cdots,n')$是树拓扑图，由树的路由规则可知，树中的任意两个计算结点之间通信所经过的结点仍在这个树内，因此对于子图G_i，V_i^C对应的计算结点之间互相通信时所经过的路由结点在G_i中对应的结点仍在该树内。根据定义7，子图G_i是一个闭合子图。

定理5 对于k叉树拓扑图G，并行度num_i的最小直径为$2\lceil\log_k num_i\rceil$。

证明：当$\log_k num_i$为整数时，并行度num_i的最小直径为$\log_k num_i+1$层k叉树的直径，即$2\log_k num_i$。当$\log_k num_i$不为整数时，并行度num_i的最小直径为$\lceil\log_k num_i\rceil+1$层$k$叉树的直径，即$2\lceil\log_k num_i\rceil$。因此，定理成立。

5 闭合最小图划分算法

基于前面的介绍，本节提出了闭合最小图划分算法，如图3所示。

输入：系统拓扑图$G=<V,E>$，
　　已占用计算结点集合$USED$，
　　新作业并行任务数num
输出：新作业分布的计算结点集合NEW
1. 计算拓扑图G中满足num并行度的最小闭合子图的模板M
2. for all $v\in V$ do
3. 　　获得以v为逻辑起点的满足模板M的子图$G'=<V',E'>$
4. 　　if $V^C\cap USED=\emptyset$ then
5. 　　$USED\leftarrow USED\cup V^C$
6. 　　输出V^C对应的计算结点集合
7. 　　end if
8. end for
9. 输出∅

图3 闭合最小图划分算法

闭合最小图划分算法的功能是基于系统的拓扑结构和已有作业所分配计算结点的信息，为新到达的作业分配对应的计算结点。该算法的基本功能如下：

首先，计算得到系统拓扑图G中满足num并行度的最小闭合子图的模板M。对于2D-Mesh来说，基于定理2和定理3，满足num并行度的最小闭合子图的模板就是一个$\lceil\sqrt{num_i}\rceil\times\left\lceil\frac{num_i}{\lceil\sqrt{num_i}\rceil}\right\rceil$的2D-Mesh；而对于一个$k$叉树状拓扑来说，基于定理4和定理5，满足$num$并行度的最小闭合子图的模板是一个$\lceil\log_k num_i\rceil+1$层$k$叉树。

其次，在系统拓扑图G中寻找一个满足模板M且不包含任何已被占用结点的子图，对应于算法第2－8行。该搜索算法可以用减枝方法进行相应的优化，这里不再赘述。

最后，如果找到满足上述模板和计算结点要求的子图，则将其对应的计算结点集合输出；否则输出空集，表示不能找到满足空间独立性且网络直径最小的空闲结点。

对于算法返回空集的情况，将根据用户的选择，或者放弃闭合子图的要求，仅在所有空闲计算结点中寻找网络直径最小的子图，并将子图中对应的计算结点分配给新作业；或者放弃网络直径最小这一目标，仅在所有空闲计算结点中寻找闭合子图（要求在当前情况下包含计算结点数最少），并将子图中对应的计算结点分配给新作业。

6 实验

我们通过MPI版的NPB测试集验证本文提出的多作业分配优化技术的效果。第6.1节介绍实验的测试用例、实验平台以及实验方法学；第6.2节对实验结果进行分析。

6.1 实验方法

6.1.1 测试用例和实验平台

我们选取NPB3.3-MPI测试集[6]作为测试用例，用于验证所提出的基于图划分的多作业分配优化方法的效果。该测试集包含5个并行Kernel（EP、MG、CG、FT、IS）和3个模拟应用（LU、SP、BT）。

我们的测试平台是2010年9月Top500榜单排名第一的Tianhe-1A超级计算机[1]的子系统。如图4所示，该子系统由两个NRM组成，每个NRM包含两个R模块。每个R模块包含8个结点，每个结点则配置了两个2.93G的6核Intel Xeon X5670 CPU和24GB RAM[11]。因此，全系统共有384个处理单元，系统使用的MPI库版本是MPICH2-1.3.1。

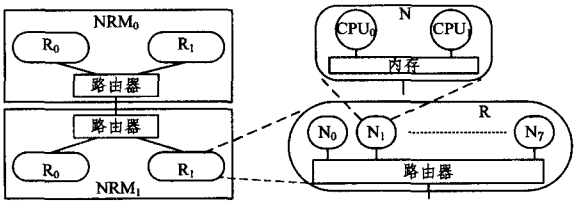

图4 测试平台体系结构

6.1.2 实验方法学

为了更好地介绍实验方法，我们首先将图4所示的测试平台中的计算结点进行编号：NRM_0的R_0模块中结点的编号为从0至7，NRM_0的R_1模块中结点的编号为从8至15，NRM_1的R_0模块中结点的编号为从16至23，NRM_1的R_1模块中结点的编号为从24至31。同时，我们还为同一个结点的处理单元（核）进行编号：CPU_0中核的编号为从0至5，

CPU_1 中核的编号为从 6 至 11。

为了验证基于闭合最小图划分模型的多作业分配优化方法的效果，我们选取目前被广泛使用的顺序作业分配法，即总将新到来的作业分配到逻辑编号较小的空闲计算结点上去（而并不管这些空闲结点之间的通信是否独立或网络直径是否最小）的方法作为基准比较对象。在测试时，测试平台可以由其他用户共享，当有其它作业在运行时，随机加载 NPB 测试集中的一个测试用例进行计算，作业分配分别采取基准顺序映射方法和本文提出的方法。通过多次测试取平均值，得到在不同分配方法下测试用例的平均执行时间。

需要指出的是，为了验证基于闭合最小图划分模型的多作业分配优化方法的可扩展性，我们在随机加载 NPB 测试用例时进行了 3 组实验，虽然 3 组实验的测试用例都是随机的，但每组实验测试用例的规模和并行度并不相同：第一组测试总是启动 4 个进程计算 A 规模的问题；第 2 组总是启动 16 个进程计算 B 规模的问题；第 3 组测试总是启动 64 个进程计算 C 规模的问题。

6.2 实验结果

如 6.1.2 节所述，实验按问题规模和并行度分为 3 组。在其他用户共享实验平台的情况下，在每组实验中随机加载 NPB 测试集中的一个测试用例进行运行。当使用基准顺序映射方案时，3 组测试中各测试用例的执行时间如表 1 所列。

表 1 基准顺序映射方案的实验结果(s)

	A.4	B.16	C.64
EP	2.332	2.458	2.452
IS	0.274	0.440	1.000
FT	1.534	7.354	13.300
CG	0.514	8.942	9.218
MG	0.680	1.242	2.530
LU	15.424	20.956	28.362
BT	20.284	24.704	25.858
SP	18.916	32.408	31.888
Avg.	7.494	12.313	14.326

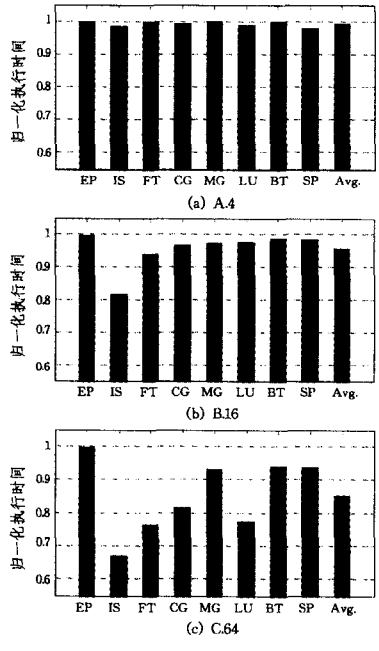

图 5 基于闭合最小图划分模型的多作业分配优化方法的效果

基于闭合最小图划分模型的多作业分配优化方法的效果如图 5 所示。与基准顺序映射方法相比，使用基于闭合最小图划分模型的多作业分配优化方法分配的作业运行时间更短，且随着问题和并行度规模的扩大，其优势越加明显。这主要是因为，对于小并行度的作业，使用基准顺序映射方法也较容易将所有进程分配到一个闭合最小子图中去，从而达到了本文提出的基于闭合最小图划分模型的多作业分配优化方法的效果；而对于大并行度的作业，上述概率就要小得多，因此基于闭合最小图划分模型的多作业分配优化方法的优势更加明显。另外，在图 5 中，对于 EP 测试用例，3 组测试的基准顺序映射方法和基于闭合最小图划分模型的多作业分配优化方法的执行时间相近，这是因为 EP 测试中几乎没有通信操作；而在所有测试用例中，基于闭合最小图划分模型的多作业分配优化方法对 IS 测试用例的效果最好，这是因为 IS 自身的执行时间较短。

结束语 本文从减少作业通信延迟和消除作业间竞争的角度，提出了一种新的多作业分配优化方法。该方法通过建立闭合最小图划分模型，将复杂抽象的多作业分配优化问题转化为具体的闭合最小图划分问题进行求解。基于“天河-1A”子系统的实验表明，该方法对于 NPB 测试集具有很好的性能优化效果。

参 考 文 献

[1] http://www.top500.org

[2] Geist A. Paving the Roadmap to EXASCALE [J]. SciDAC Review, NUMBER 16 Special Issue, 2010

[3] Jose A P, Jose M A, Jose A L. Optimization-based Mapping Framework for Parallel Applications [J]. Journal of Parallel and Distributed Computing, 2011, 10(71): 1377-1387

[4] Kumar V. Introduction to Parallel Computing (2nd Ed.) [M]. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002

[5] Sinnen O, Sousa L A. Communication Contention in Task Scheduling [J]. IEEE Transactions on Parallel and Distributed Systems, 2005, 16(6): 503-515

[6] http://www.nas.nasa.gov/publications/npb.html

[7] 李秋阳. 并行系统性能评估技术研究[D]. 天津：南开大学，2002

[8] Kim J, Dally W J, Scott S, et al. Technology-Driven, Highly-Scalable Dragonfly Topology [J]. SIGARCH Comput. Archit. News, 2008, 36: 77-88

[9] Moadeli M, Shahrabi A, Vanderbauwhede W, et al. Communication Modelling of the Spidergon NoC with Virtual Channels [C]// Proceedings of the 2007 International Conference on Parallel Processing. Washington, DC, USA, 2007: 76-76

[10] 杨晓东，陆松，牟胜梅. 并行计算机体系结构—技术与分析[M]. 北京：科学出版社，2009

[11] Yang X J, Liao X K, Lu K, et al. The TianHe-1A Supercomputer: Its Hardware and Software [J]. Journal of Computer Science and Technology, 2011, 26(3): 344-351

[12] 王之元. 并行计算可扩展性分析与优化：能耗、可靠性与计算性能[D]. 长沙：国防科学技术大学，2011