第 31 卷 第 1 期
2008 年 1 月
计 算 机 学 报
CHINESE JOURNAL OF COMPUTERS
Vol. 31 No. 1
Jan. 2008

一类非规则并行应用问题的通信集生成算法

胡长军 李 静 王 珏 姚广利 李永红 丁 良 李建江
（北京科技大学信息工程学院 北京 100083）

摘 要 非规则计算是大规模并行应用中普遍存在和影响效率的关键问题. 在基于分布式内存的数据并行范例中，如何针对非规则数组引用，有效地生成本地内存访问序列和通信集，是并行编译生成 SPMD 结点程序所必须解决的重要问题. 文中针对两重嵌套循环中，下一层循环边界是上一层循环变量的线性或非线性函数，数组下标是两层循环变量的非线性函数这样一类包含非规则数组引用的并行应用问题，提出了一种在编译时生成通信集的代数算法. 并且针对 cyclic(k)数据分布和线性对齐模板，借助整数格概念，给出了编译时全局地址和本地地址之间的转换方法. 文中还给出了相应的经过通信优化的 SPMD 结点程序. 最后通过实例验证了算法的正确性. 该算法的意义在于避免了传统 Inspector/Executor 非规则计算模型中的 Inspector 阶段，从而节省了运行时 Inspector 阶段通过穷举下标生成通信集的巨大开销.

关键词 非规则计算；通信集生成；并行编译；通信优化；SPMD
中图法分类号 TP311

Communication Set Generation for a Special Case of Irregular Parallel Applications

HU Chang-Jun LI Jing WANG Jue YAO Guang-Li LI Yong-Hong DING Liang LI Jian-Jiang
(*School of Information Engineering, University of Science and Technology Beijing, Beijing* 100083)

Abstract Irregular computing significantly influences the performance of large scale parallel applications. How to generate local memory access sequence and communication set efficiently for irregular array reference is an important issue in compiling a data-parallel language into a single program multiple data (SPMD) code for distributed-memory machines. By far, many researches have focused on the problem of communication set generation under regular array references in parallel loops. However, little researches give attentions to generating communication set for irregular array accesses in loop nests. In general case of irregular accesses, Inspector/Executor model is adopted to scan over array elements at Inspector phase of run time such that communication set can be constructed. This paper proposes an approach to derive an algebraic solution of communication set enumeration at compile time for the situation of irregular array references in nested loops. And this paper introduces integer lattice into alignment and cyclic(k)-distribution for global-to-local and local-to-global index translations. Then it also presents the algorithm for the corresponding SPMD code generation. When the parameters of alignments and cyclic(k) and array references are known, the SPMD code can then be completely derived at compile time such that no inspector-like run-time code will be needed to construct enumeration of the communication set.

收稿日期：2006-04-21；最终修改稿收到日期：2007-10-09. 本课题得到国家“八六三”高技术研究发展计划项目基金(2006AA01Z105)、国家自然科学基金(60373008)和教育部重点基金(106019)资助. **胡长军**，男，1963 年生，博士，教授，博士生导师，主要研究领域为并行计算与并行编译技术、并行软件工程、网络存储体系结构、数据工程与软件工程. **李 静**，女，1982 年生，硕士研究生，主要研究方向为并行计算与并行编译技术. **王 珏**(通信作者)，男，1981 年生，博士研究生，主要研究方向为并行计算与并行编译技术. E-mail：ncepu5@163.com. **姚广利**，男，1982 年生，硕士研究生，主要研究方向为并行计算与并行编译技术. **李永红**，女，1982 年生，硕士研究生，主要研究方向为并行计算与并行算法. **丁 良**，男，1980 年生，硕士研究生，主要研究方向为并行计算与并行算法. **李建江**，男，1971 年生，博士，副教授，主要研究方向为并行计算、并行编译与多线程技术.

Keywords　irregular computing；communication set generation；parallelizing compilers；communication optimization；SPMD scheme

1　引　言

数据并行语言如 HPF 等以全局名空间、松同步结构、单线程控制的特点极大地减轻了并行编程难度，但是却将数据分布、计算分布、本地地址列举、通信集生成、消息打包和消息拆包等任务留给了并行编译系统. 其中，在给定数据分布和迭代中数组引用模式的前提下，为每个结点生成通信集和结点程序，是并行编译技术的关键. 对于循环边界是常数，并且数组下标是循环变量的线性函数，这种规则并行应用在编译时的通信集生成问题，人们已经做过大量研究. 这方面的研究主要有：Koelbel 和 Mehrotra[1] 首先提出用 closed-form 表示通信集；Gupta 等人[2] 对数组按 block 或 cyclic 分布的情况，使用 closed-form 来表示通信集；Chatterjee 等人[3] 使用有限状态机（FSM）理论列举出对本地内存地址的访问序列；Adve 和 Mellor-Crummey[4] 在 Rice dHPF 编译器中使用整数集合框架来表示循环划分和通信集生成；Kennedy 等人[5-6] 采用整数格方法产生内存访问序列；Tseng 和 Gaudiot[7] 在整数格中使用 Smith-Normal-Form 分析得到通信集列举的一个代数解；Lee 和 Chen[8] 提出了一种将整数格和集合论结合在一起的通信集生成方法；Huang 和 Shiu[9] 通过研究一个处理器上两个有相同偏移量和目的地的活动元素之间的本地块距离，分别对发送和接受阶段给出了不同的通信集生成算法；Hwang[10] 针对多维数组以及数组下标是仿射或者经过轴变换的情况，提出了一种不依赖于解线性 diophantine 方程的通信集生成方法.

但是，对于两重嵌套循环中，下一层循环边界是上一层循环变量的线性或非线性函数，并且数组下标表达式是两层循环变量的非线性函数，这样涉及到非规则数组引用的通信集生成问题，研究成果却很少. 实际上，这样的问题在并行应用领域十分常见. 图 1 是从 Perfect Benchmarks 的 TRFD 代码中截取的一段代码. 在这段代码中，第二层循环变量 j 的上界是第一层循环变量 i，数组 A 和 B 的引用函数是非线性函数，分别为 $i*(i-1)/2+j$ 和 $j*(j-1)/2+i$. 因为在 LHS（Left Hand Side）和 RHS（Right Hand Side）的数组全局地址之间没有仿射关系，所以普通针对仿射的通信集生成技术并不适用于这种非规则问题.

```
DO i=1, N
  DO j=1, i
  A(i*(i-1)/2+j)=F(B(j*(j-1)/2+i))
  …
  ENDDO
ENDDO
```

图 1　有非规则数组引用的嵌套循环

对于非规则数组引用的通信集生成问题，目前主要是通过采用 Inspector/Executor 非规则计算模型的运行支持库来解决. Inspector/Executor 非规则计算模型分为两个阶段，分别是 Inspector 阶段和 Executor 阶段，前者分析数组引用并生成通信调度，后者使用通信调度进行通信并完成计算. Inspector 阶段的主要任务是进行全局地址和本地地址之间的转换，生成本地内存访问序列，并且通过穷举下标来生成通信集，因此 Inspector 阶段开销很大. 这方面的主要研究有：由 Maryland 大学于 1994 年完成的 CHAOS/PARTI 库[11]，该库最早提出了支持非规则并行计算的 Inspector/Executor 模型；PILAR[12] 是由 Illinois 大学开发的非规则库，主要针对混合型数组下标问题；LPARX[13] 是一个 C++ 的运行支持库，支持自动按 block 划分的非规则问题；Guo[14] 等人提出了一种使通信量最小化的数组分布模式，并且使用符号分析技术产生非规则数组引用的通信集. 这些研究工作中一个不可避免的问题是：运行时 Inspector 阶段开销很大，非规则并行应用问题难以取得高效率. 特别是对于多重循环中的非规则数组引用问题，Inspector/Executor 模型就更加无能为力.

本文针对两重嵌套循环中，下一层循环边界是上一层循环变量的线性或非线性函数，并且数组下标表达式是两层循环变量的非线性函数，这样包含非规则数组引用的通信集生成问题，提出了一种使用代数不等式来表示循环边界、数组引用和数组分布的约束条件，通过解代数不等式组在编译时得到非规则通信集的一个代数解的 AIS（Algebraic Inequations Solver）算法. 其次，本文还给出了相应的经过通信优化的 SPMD 结点程序的生成方法：通过将执行只需要本地数据的循环迭代和接收从其它处理器发来的数据这两个阶段重叠，来隐藏通信延迟；

并且在发送消息阶段，为了避免多个处理器同时向同一个处理器发送消息时可能会出现的冲突情况，通过取模方法将各个处理器发送消息的顺序岔开. 另外，本文还在 cyclic(k)数据分布和线性对齐模板中借助整数格概念，给出了编译时全局地址和本地地址之间的转换方法. 使用 AIS 算法及其相应的 SPMD 结点程序，当得到数据分布以及迭代的所有参数后，在编译时就可以完成全局地址和本地地址之间的转换，生成本地内存访问序列和通信集，从而得到完整的 SPMD 结点程序，因此避免了传统 Inspector/Executor 非规则计算模型中的 Inspector 阶段，从而节省了运行时 Inspector 阶段通过穷举下标生成通信集的巨大开销.

本文第 2 节详细介绍了 AIS 算法及其相应的 SPMD 结点程序；第 3 节通过实际数据的测试结果表明，AIS 算法是正确和高效的，同时还具有良好的可扩展性；最后是结论部分.

2 AIS 算法及其相应的 SPMD 结点程序

下面分别介绍全局地址/本地地址的转换方法、AIS 算法及其相应的经过通信优化的 SPMD 结点程序. 图 2 是 AIS 算法的研究对象：第 1 行～第 7 行给出了数据分布、线性对齐模板和处理器个数；第 8 行～第 10 行给出了循环边界和非规则数组引用模式；计算分布采用拥有者计算原则. AIS 算法的目的是使用代数不等式来表示循环边界、数组引用和数组分布的约束条件，通过解代数不等式组在编译时得到通信集的一个代数解.

```
1. real A(0:(n_A−1)),B(0:(n_B−1))
2. !processor P(0:(n_p−1))
3. !template T_A,T_B
4. !distribute T_A(cyclic(m_A)) onto P
5. !distribute T_B(cyclic(m_B)) onto P
6. !alignment A(i) with T_A(a_A * i+b_A)
7. !alignment B(i) with T_B(a_B * i+b_B)
8. for (i_g=0:(n_g−1))
9. for (j_g=s_1 · i_g+t_1:s_2 · i_g+t_2)
10. A(α_A1 · i_g^2+β_A1 · i_g+γ_A1+α_A2 · j_g^2+β_A2 · j_g+γ_A2)=
    B(α_B1 · i_g^2+β_B1 · i_g+γ_B1+α_B2 · j_g^2+β_B2 · j_g+γ_B2)
```

图 2 包含非规则数组引用的数据并行程序

2.1 全局地址/本地地址的转换方法

如图 2 所示，程序声明了两个一维数组 A 和 B，它们分别根据模板 T_A 和 T_B 在处理器数组 P 上进行分布. 第 4 条语句中，cyclic(m_A)表示将模板 T_A 的 m_A 个元素同时分配给处理器数组 P 中的一个处理器. 在各个处理器上分布的模板 T_A 元素可以被看成一个 $r-c$ 整数格. 其中，r 表示某个处理器上分布的 m 个模板 T_A 元素的行；c 表示这 m 个元素所在的列. 第 6 条语句表示将元素 $A(i)$ 和元素 $T_A(a_A \cdot i+b_A)$ 分布到同一个处理器上，因此在各个处理器上分布的数组 A 元素在这个 $r-c$ 整数格中也有自己相应的坐标.

假设某个数组 A 元素的全局下标是 i_A，在 $r-c$ 格中的局部下标是 (r_A, c_A)，根据数据分布情况，可以得到

$$(r_A, c_A)=\left(\left\lfloor \frac{a_A \cdot i_A+b_A}{m_A \cdot n_p} \right\rfloor, (a_A \cdot i_A+b_A)\%m_A)\right) \tag{1}$$

其中，n_p 表示处理器个数，m_A 表示模板 T_A 的 cyclic 分布参数，a_A 和 b_A 表示数组 A 与模板 T_A 之间的线性对齐参数.

相似地，假设某个数组 B 元素的全局下标是 i_B，在 $r-c$ 格中的局部下标是 (r_B, c_B)，根据数据分布情况，可以得到

$$(r_B, c_B)=\left(\left\lfloor \frac{a_B \cdot i_B+b_B}{m_B \cdot n_p} \right\rfloor, (a_B \cdot i_B+b_B)\%m_B\right) \tag{2}$$

其中，n_p 表示处理器个数，m_B 表示模板 T_B 的 cyclic 分布参数，a_B 和 b_B 表示数组 B 与模板 T_B 之间的线性对齐参数.

通过在 cyclic(k)数据分布和线性对齐模板中引入整数格概念，我们得到了一种编译时全局地址和本地地址之间的转换方法，转换公式如式(1)和(2)所示. 这种转换方法计算简单，而且引入整数格概念能够更好地表现 cyclic(k)数据分布和线性对齐模板的特点.

2.2 通信集生成的 AIS 算法

如图 2 所示，程序中有两层循环，其中第二层循环 j_g 的边界是第一层循环变量 i_g 的线性函数，分别为 $s_1 \cdot i_g+t_1$ 和 $s_2 \cdot i_g+t_2$；数组引用函数是循环变量 i_g 和 j_g 的非线性函数，数组 A 的引用函数 I_A 和数组 B 的引用函数 I_B 分别为

$$I_A=\alpha_{A1} \cdot i_g^2+\beta_{A1} \cdot i_g+\gamma_{A1}+\alpha_{A2} \cdot j_g^2+\beta_{A2} \cdot j_g+\gamma_{A2} \tag{3}$$

$$I_B=\alpha_{B1} \cdot i_g^2+\beta_{B1} \cdot i_g+\gamma_{B1}+\alpha_{B2} \cdot j_g^2+\beta_{B2} \cdot j_g+\gamma_{B2} \tag{4}$$

之所以这样设定循环边界和数组引用函数的形式，是为了保证一定可以得到通信集的一个代数形

式解. 实际上，只要能给出所列代数不等式组的解的代数形式，循环边界和数组引用函数的形式并不仅限于此. 但是，如果不能给出解的代数形式，那么就需要通过类似空间探测[15]的方法来得到通信集列举，而这种方法的时间和空间复杂度都很高.

根据数据分布情况，可以得到

$$I_{A1} \le I_A \le I_{A2} \tag{5}$$

其中，$I_{A1} = \left\lceil \frac{r_A \cdot m_A \cdot n_p + m_A \cdot p - b_A}{a_A} \right\rceil$，$I_{A2} = \left\lfloor \frac{r_A \cdot m_A \cdot n_p + m_A \cdot p + m_A - 1 - b_A}{a_A} \right\rfloor$，$p$ 表示处理器 p 的处理器号，n_p表示处理器个数，r_A表示数组 A 元素在 $r-c$ 格中的行坐标.

$$I_{B1} \le I_B \le I_{B2} \tag{6}$$

其中，$I_{B1} = \left\lceil \frac{r_B \cdot m_B \cdot n_p + m_B \cdot q - b_B}{a_B} \right\rceil$，$I_{B2} = \left\lfloor \frac{r_B \cdot m_B \cdot n_p + m_B \cdot q + m_B - 1 - b_B}{a_B} \right\rfloor$，$q$ 表示处理器 q 的处理器号，n_p表示处理器个数，r_B表示数组 B 元素在 $r-c$ 格中的行坐标.

根据循环边界范围，可以得到

$$0 \le i_g \le n_g - 1 \tag{7}$$

$$s_1 \cdot i_g + t_1 \le j_g \le s_2 \cdot i_g + t_2 \tag{8}$$

下面通过解由式(5)～(8)组成的代数不等式组来计算出通信集的一个代数解，具体过程如下.

由式(3)，(5)，(7)和(8)，根据代数不等式的性质，可以计算出 i_g的范围 k_1；相似地，由式(4)，(6)，(7)和(8)，可以计算出 i_g的范围 k_2，则 i_g的范围 k_i 为

$$k_i = \{k_1 \cap k_2\} \tag{9}$$

由式(3)，(5)，(8)和(9)，根据函数单调性，可以计算出 j_g 的范围 k_3；相似地，由式(4)，(6)，(8)和(9)，可以计算出 j_g的范围 k_4，则 j_g的范围 k_j 为

$$k_j = \{k_3 \cap k_4\} \tag{10}$$

其中，范围 k_i和 k_j都是 r_A，r_B，p 和 q 的函数.

根据数据分布情况，可以得到 r_A和 r_B的范围分别为

$$0 \le r_A \le \left\lfloor \frac{a_A \cdot (n_A - 1) + b_A}{m_A \cdot n_p} \right\rfloor \tag{11}$$

$$0 \le r_B \le \left\lfloor \frac{a_B \cdot (n_B - 1) + b_B}{m_B \cdot n_p} \right\rfloor \tag{12}$$

根据式(9)～(12)，将 r_A和 r_B值进行任意组合并代入范围 k_i和 k_j，得到范围 k'_i和k'_j(其中，范围 k'_i和 k'_j是 p 和 q 的函数). 将范围 k'_i和 k'_j中的所有 i_g和 j_g的值进行一一组合，得到与处理器 q 发送给处理器 p 的通信集相对应的(i_g，j_g)集合，表示为

$$Comm_Set(i_g, j_g)_{[p,q]} = \{(i_g, j_g) \mid i_g \in k'_i \cap j_g \in k'_j\} \tag{13}$$

将式(13)代入式(4)中，得到处理器 q 发送给处理器 p 的通信集(数组 B 元素的集合)，表示为

$$Comm_Set(B)_{[p,q]} = \{I_B(i_g, j_g) \mid (i_g, j_g) \in Comm_Set(i_g, j_g)_{[p,q]}\} \tag{14}$$

将式(13)代入式(3)中，结果就是处理器 p 上需要引用$Comm_Set(B)_{[p,q]}$的数组 A 元素的集合，表示为

$$Comm_Set(A)_{[p,q]} = \{I_A(i_g, j_g) \mid (i_g, j_g) \in Comm_Set(i_g, j_g)_{[p,q]}\} \tag{15}$$

根据式(2)和(14)，可以得到与 $Comm_Set(B)_{[p,q]}$相对应的 $r-c$ 格坐标集合为

$$Comm_Set(B:r-c)_{[p,q]} = \{r_B = \left\lfloor \frac{a_B \cdot I_B + b_B}{m_B \cdot n_p} \right\rfloor, c_B = (a_B \cdot I_B + b_B) \% m_B \mid I_B \in Comm_Set(B)_{[p,q]}\} \tag{16}$$

相似地，根据式(1)和(15)，得到与 $Comm_Set(A)_{[p,q]}$相对应的 $r-c$ 格坐标集合为

$$Comm_Set(A:r-c)_{[p,q]} = \{r_A = \left\lfloor \frac{a_A \cdot I_A + b_A}{m_A \cdot n_p} \right\rfloor, c_A = (a_A \cdot I_A + b_A) \% m_A \mid I_A \in Comm_Set(A)_{[p,q]}\} \tag{17}$$

到此为止，我们就得到了非规则数组引用的通信集的一个代数解. 其中，式(16)表示处理器 q 要发送给处理器 p 的通信集(数组 B 元素的集合)；式(17)表示处理器 p 上和通信集相对应的数组 A 元素集合. 解由式(5)～(8)组成的代数不等式组时，根据参数不同，要分成很多种情况进行讨论. 但是，在实际应用中，因为数据分布以及迭代中的各个参数都是确定的，所以只需要计算其中一种情况，计算量很小.

2.3　SPMD 结点程序的生成方法

在分布式内存机器上，每个处理器上执行的编译代码都是 SPMD 形式的. 在 SPMD 模型中，每个处理器都要经历以下 4 个阶段：发送其它处理器需要的本地数据、执行只需要本地数据的循环迭代、接收从其它处理器发来的数据和执行需要远程数据的循环迭代. 因为本地计算部分(第 2 阶段)在接收阶段(第 3 阶段)之前执行，并且每个处理器都首先发送数据(第 1 阶段)，所以就可以通过将第 2 阶段和第 3 阶段重叠来隐藏通信延迟. 在发送消息过程中，为了避免多个处理器同时向同一个处理器发送消息时可能会出现的冲突情况，通过取模的方法，使用 $i \in \{0, 1, \cdots, n_p - 1\}$，$q = (i + p) \bmod n_p$语句将各个

处理器发送消息的顺序岔开.

在发送处理器 q 上，$Comm_Set(B:r-c)_{[p,q]}$ 消息打包序列是由 $Comm_Set(i_g,j_g)_{[p,q]}$ 产生的. 在接收处理器 p 上，对消息中数组 B 元素的非本地引用序列也是通过 $Comm_Set(i_g,j_g)_{[p,q]}$ 产生的. 因为两个序列都是由同一个 $Comm_Set(i_g,j_g)_{[p,q]}$ 产生的，所以接收处理器将以发送处理器上打包的序列来访问这个消息.

基于以上理论，可以得到与图 2 中数据并行程序相对应的 SPMD 结点代码，如图 3 所示.

```
1. for p=0 to n_p-1 do
2.
3.  /* pack and send messages to other processors */
4.    for i∈{0,1,…,n_p-1} do
5.      q=(i+p) mod n_p and p≠q
6.      calculate Comm_Set(i_g,j_g)[q,p]
7.      for (i_g,j_g)∈Comm_Set(i_g,j_g)[q,p]
8.        Append Comm_Set(B:r-c)[q,p] to buf_send(q)
9.        Send buf_send(q) to processor if not empty
10.       end
11.     enddo
12.
13. /* perform local iteration */
14.   calculate Comm_Set(i_g,j_g)[p,p]
15.   for(i_g,j_g)∈Comm_Set(i_g,j_g)[p,p]
16.     Comm_Set(A:r-c)[p,p]=Comm_Set(B:r-c)[p,p]
17.   end
18.
19. /* receive data and perform remaining iterations */
20.   while expecting messages
21.   wait for a message from q and store into Buf[q]
22.     s=0
23.     calculate Comm_Set(i_g,j_g)[p,q]
24.     for(i_g,j_g)∈Comm_Set(i_g,j_g)[p,q]
25.       Comm_Set(A:r-c)[p,q]=Buf[q](s)
26.       s=s+1
27.     end
28.   end
29.
30. enddo
```

图 3 与图 2 相对应的 SPMD 结点程序

3 测试结果与分析

为了验证本文提出的 AIS 算法的正确性和高效性及其良好的可扩展性，我们实现了 AIS 算法，并在我们之前开发的扩展 OpenMP 编译器[16-17]中嵌入了该算法. 下面以 Perfect Benchmark 中 TRFD 的子程序 OLDA 的程序片断（如图 4 所示）以及三次采油油藏数值模拟软件原代码文件 engbal.f 的程序片断（如图 5 所示）为例，分别进行了两组对比实验，实验获得了很好的效果. 第一组实验是在数据量一定而结点数变化的情况下，比较分别使用 AIS 算法和传统的 Inspector/Executor 非规则计算模型中的穷举算法的通信集生成时间（如图 5 所示）；第二组实验是在数据量一定而结点数变化的情况下，比较分别使用 AIS 算法和传统的 Inspector/Executor 非规则计算模型中的穷举算法的整个并行程序的运行时间（如图 6 所示）.

```
DO mrs=0, (N*N+N)/2-1
 DO mi=0, N-1
  DO mj=0, mi-1
   Mrsij=(mi*mi+mrs*(N*N+N))/2+mj+1
   Xrsij(mrsij)=xij(mj)
  END DO
 END DO
END DO
```

图 4 Perfect Benchmark 中 TRFD 的子程序 OLDA 片断

```
DO M=1, NWELL
 DO IWB=1, M
  IJK=(M*M+M)/2+IWB
  IER=IWB*IWB
  TTCHG(IJK)=BCBF(IER)
 END DO
END DO
```

图 5 三次采油油藏数值模拟软件原代码文件 engbal.f 的程序片断

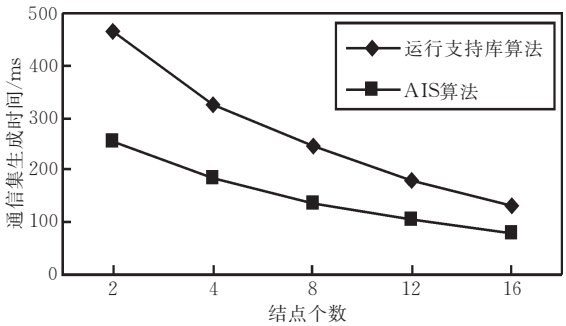

图 6 图 4 程序对应的通信集生成时间比较（$N=80$，即循环迭代次数为 10^7）

实验测试平台为：共享内存的 USTB SMP-Cluster 并行计算机，其中共有 16 个结点，每结点配置两个 Intel Xeon 3.0GHZ/1024K L2 Cache 处理器，一个 2GB 内存，结点之间通过千兆以太网互联；操作系统为 Redhat Linux 9.0，内核版本 2.4.21-smp，串行编译器为 gcc 3.3.2，MPI 运行环境为 Argonne 实验室开发的mpich-1.2.7，由 $MPI_Wtime()$ 函数来测试执行时间.

如图 6 和图 7 所示，第一组实验表明，单纯从通信集生成算法来看，本文中提出的 AIS 算法和采用 Inspector/Executor 非规则计算模型的传统运行支持库算法相比，在性能上有接近 50% 的显著提高. 这一方面是因为 AIS 算法的复杂度比传统穷举算法要

低很多，另一方面是因为 AIS 算法节省了传统算法中所必须的各个结点间交换通信集信息的通信时间.

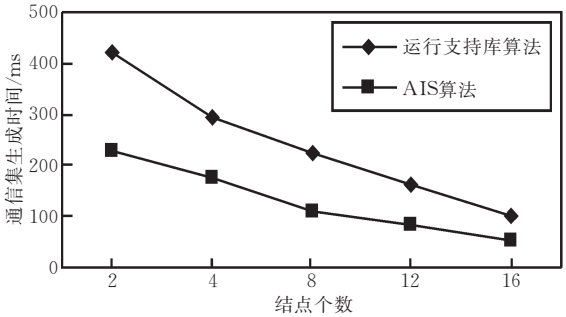

图 7　图 5 程序对应的通信集生成时间比较（$NWELL=5000$，即循环迭代次数为 10^7）

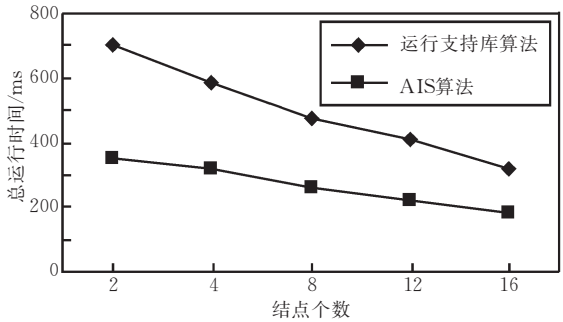

图 8　图 4 程序对应的总运行时间比较（$N=80$，即循环迭代次数为 10^7）

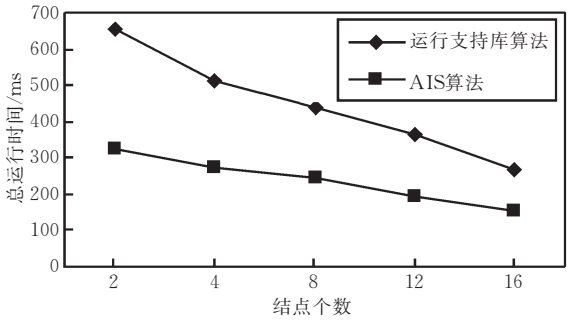

图 9　图 5 程序对应的总运行时间比较（$NWELL=5000$，即循环迭代次数为 10^7）

如图 8 和图 9 所示，从第二组实验中可以看到，由于在运行时避免了传统 Inspector/Executor 非规则计算模型中的 Inspector 阶段，因此使用 AIS 算法的并行程序的运行时间与使用传统运行支持库方法相比，在效率上有了非常显著的提高. 但同时也看到，总运行时间并没有随着结点数的增加而成比例地减少，这是因为随着结点数的增加，各个结点之间的通信也增加了.

4　结　论

到目前为止，对非规则数组引用的通信集生成问题，主要还是通过采用 Inspector/Executor 非规则计算模型的运行支持库来解决. 虽然国内外研究人员一直致力于改进 Inspector/Executor 模型，但是他们都不能从根本上解决运行时 Inspector 阶段开销很大. 非规则并行应用问题难以取得高效率这个重大缺陷. 本文针对非规则数组引用的通信集生成问题，提出了一种在编译时通过代数方法生成非规则通信集的新思路，避免了传统 Inspector/Executor 非规则计算模型中的 Inspector 阶段，从而节省了运行时 Inspector 阶段通过穷举下标生成通信集的巨大开销. 为研究解决非规则计算问题增加了一条新途径.

参　考　文　献

[1] Koelbel C. Compile-time generation of regular communications patterns//Proceedings of the Conference on High Performance Networking and Computing, Albuguerque. New Mexico, United States, 1991: 101-110

[2] Gupta S K S, Kaushik S D, Huang C H, Sadayappan P. Compiling array expressions for efficient execution on distributed-memory machines. Journal of Parallel and Distributed Computing, 1996, 32(2): 155-172

[3] Chatterjee S, Gilbert J R, Long F J E, Schreiber R, Teng S H. Generating local addresses and communication sets for data-parallel programs. Journal of Parallel and Distributed Computing, 1995, 26(1): 72-84

[4] Adve V, Mellor-Crummey J. Using integer sets for data-parallel program analysis and optimization//Proceedings of the Conference on Programming Language Design and Implementation. Montreal, Quebec, CA, 1998: 186-198

[5] Kennedy K, Nedeljkovic N, Sethi A. Efficient address generation for block-cyclic distributions//Proceedings of the International Conference on Supercomputing. Barcelona, Spain, 1995: 180-184

[6] Kennedy K, Nedeljkovic N, Sethi A. A linear-time algorithm for computing the memory access sequence in data-parallel programs//Proceedings of the ACM SIGPLAN Symposium on Principles and Practices of Parallel Programming. Santa Barbara, Santa Barbara, California, United States, 1995: 102-111

[7] Tseng E H Y, Gaudiot J L. Communication generation for aligned and Cyclic(k) distributions using integer lattice. IEEE Transactions on Parallel Distributed Systems, 1999, 10(2): 136-146

[8] Lee P Z, Chen W Y. Generating communication sets of array assignment statements for block-cyclic distribution on distributed memory parallel computers. Parallel Computing, 2002, 28(9): 1329-1368

[9] Huang T C, Shiu L C. Efficient communication sets generation for block-cyclic distribution on distributed-memory ma-

chines. Journal of Systems Architecture, 2003, 48: 255-265

[10] Hwang G H. An efficient algorithm for communication set generation of data parallel programs with block-cyclic distribution. Parallel Computing, 2004, 30(4): 473-501

[11] Saltz J, Ponnusamy R, Sharma S D, Moon B, Hwang Y S, Uysal M, Das R. A manual for the chaos runtime library. Department of Computer Science, University of Maryland, Maryland, USA: Technical Report CS-TR3437, 1995

[12] Lain A, Banerjee P. Compiler support for hybrid irregular accesses on multicomputers//Proceedings of the International Conference on Supercomputing. Philadelphia, Pennsylvania, United States, 1996: 1-9

[13] kohn S R, Baden S B. A robust parallel programming model for dynamic non-uniform scientific computations//Proceedings of the Scalable High-Performance Computing Conference. Los Alamitos, 1994: 509-517

[14] Guo M Y, Pan Y, Lin Z. Symbolic communication set generation for irregular parallel applications. The Journal of Supercomputing, 2003, 25(3): 199-214

[15] Ancourt C, Irigoin F. Scanning polyhedra with do loops//Proceedings of the Principles and Practice of Parallel Programming. Williamsburg, Virginia, United States, 1991: 39-50

[16] Hu C J, Wang J, Li J J. Language support for multi-paradigm and multi-grain parallelism on smp-cluster. International Journal of Computers and Applications, 2005, 2: 196-203

[17] Wang J, Hu C J, Lai J X, Zhao Y D, Zhang S Q. Multi-paradigm and Multi-grain Parallel Execution Model Based on SMP-Cluster//Proceedings of the IEEE 2006 John Vincent Atanasoff International Symposium on Modern Computing. Sofia, Bulgaria, 2006: 266-272

HU Chang-Jun, born in 1963, Ph.D., professor, Ph. D. supervisor. His main research interests include parallel computing, parallel compilation technology, parallel software engineering, network storage system, data engineering and software engineering.

LI Jing, born in 1982, master. His research interests include parallel computing and parallel compilation technology.

WANG Jue, born in 1981, Ph. D. candidate. His research interests include parallel computing and parallel compilation technology.

YAO Guang-Li, born in 1982, master. His research interests include parallel computing and parallel compilation technology.

LI Yong-Hong, born in 1982, master. His research interests include parallel computing and parallel algorithm.

DING Liang, born in 1980, master. His research interests include parallel computing and parallel algorithm.

LI Jian-Jiang, born in 1971, Ph. D., associate professor. His main research interests include parallel computation, parallel compilation and multi-threaded technology.

Background

The work in this paper belongs to communication set generation for irregular applications. Up to now, a lot of researches have focused on the problem of communication set generation under regular array references in parallel loops. However, little researches give attentions to generate communication set for irregular array accesses in loop nests. In general case of irregular accesses, Inspector/Executor model is adopted to scan over array elements at Inspector phase of run-time such that communication set can be constructed. The contributions of this work include:

(1) Proposing an approach to derive an algebraic solution of communication set enumeration at compile time for the situation of irregular array references in nested loops;

(2) Introducing integer lattice into alignment and cyclic (k)-distribution for global-to-local and local-to-global index translations;

(3) Presenting the authors' algorithm for the corresponding SPMD code generation. So when the parameters of alignments and cyclic(k) and array references are known, the SPMD code can then be completely derived at compile time such that no inspector-like run-time code will be needed to construct enumeration of the communication set.

This work is supported by The National High Technology Research and Development Program (863 Program) under grant No. 2006AA01Z105, the National Natural Science Foundation of China under grant No. 60373008 and the Ministry of Education of the People's Republic of China Major Foundation under grant No. 106019. The objective of the National High Technology Research and Development Program (863 Program) is to improve the productivity of parallel programmer using parallel compiling techniques. The National Natural Science Foundation of China pays attentions to OpenMP extension on clusters. The Ministry of Education of the People's Republic of China Major Foundation focuses on a difficult problem in parallel computing, i. e., out-of-core irregular problem. In communication scheduling area, The authors have improved efficiency of collective communication using compiling techniques. This paper proposes a communication generation approach, which integrates the benefits from the algebra method and the integer lattice method, to derive an algebraic solution of communication set enumeration.