CME 分析中的丢番图方程求解*)

Diophantine Equation Solution in CME Analysis

舒 辉 康 绯

(信息工程大学信息安全学院 郑州450002)

Abstract CME(Cache Miss Equation)is a kind of framework of cache miss rate analysis of loops, in practice sampling test of loop iterations is used in CME analysis, and the key problem of sampling test is how to decide whether there exists integer solution of Diophantine equation in limited conditions. This paper gives a general description of this problem, provides an algorithm called Lattice Test to solve it.

Keywords CME analysis, Diophantine equation, Lattice test

1. 引言

循环的 Cache 不命中率分析是编译优化中的关键技术之一。CME(Cache Miss Equation)是美国 Princeton 大学的 S. Ghosh 博士提出的循环 Cache 不命中率分析的计算模型[1],它以循环中数组引用的重用向量分析为基础,根据 Cache 数据映射的规则用约束条件下丢番图方程对数组引用在不同重用向量上的 Cache 冲突情况进行精确的描述(一般为线性约束条件),明确了方程整数解的数目与 Cache 不命中次数之间的关系,为较为精确地分析循环的 Cache 不命中率奠定了基础。我们把通过 CME 来分析 Cache 不命中率的过程简称为 CME 分析。由于 CME 本质上是一个约束条件下的丢番图方程,而求约束条件下丢番图方程整数解数目是一个 NP 问题,直接计算的代价太大,因而在实际中 S. Ghosh 提出了用抽样的方法,通过在循环迭代空间中随机抽取一定比例的迭代点,分析在这些抽样的迭代点上数组引用的数据访问是否 Cache 命中(也即抽样检验),并以此推断循环整体的 Cache 不命中率次数,这种抽样分析的方法在实践中已被证明是可行的,但通过 CME 分析数组引用的数据访问在单个迭代点上是否发生 Cache 不命中,对于直接映射的 Cache 而言,其本质上是一个判定约束条件下丢番图方程是否存在整数解的问题,这在理论上仍然是一个 NP 问题,对此 S. Ghosh 并没有给出相应的计算方法。

关于判定丢番图方程在线性约束条件下是否存在整数解的问题也是并行识别中相关性测试所要重点解决的问题,对此比较著名的有 GCD 测试、Power 测试、Omega 测试以及 Lamda 测试等,这些测试(除 Omega 测试外)都属于保守性的数学测试方法(即只给出了丢番图方程无整数解的必要条件),而 Omega 测试则比较适用于丢番图方程组的分析,并且对于一些简单的丢番图方程都存在着代价过大的情况,这些都不符合 CME 抽样检验中较为快速、精确地判断丢番图方程整数解存在性的要求。

本文对 CME 抽样分析中丢番图方程整数解判定问题作了一般性描述,利用传统的相关性测试中用到的一些数学方法(包括傅立叶-密西根变量削减算法[2]、Ceiling and Flooring[3]),并结合较新的整数计算理论(L^3格约化算法[4])给出了线性约束条件下多变量 CME 整数解存在性的判定算法——格测试算法,并给出了具体的分析实例和相关的部分试验数据。

2. CME 分析与丢番图方程整数解的判定

循环中两个数组引用 R_A 和 R_B 之间在某重用向量上的 Cache 冲突 CME 的通用形式如下:

$$\begin{cases} ML_{R_A}(\vec{i}) = ML_{R_B}(\vec{j}) + n * C_s/k + b \\ -L_{off} < b \leqslant L - 1 - L_{off}, L_{off} = ML_{R_A}(\vec{i}) \bmod L, n \neq 0 \\ \vec{j} \in [\vec{p}, \vec{i}), \vec{p} = \vec{i} - \vec{r} \end{cases}$$

其中向量 $\vec{i}$ 和 $\vec{j}$ 为循环迭代空间中的迭代点,C_s 为 Cache 的容量,L 为 Cache 的行长,k 为 Cache 的相联数,$\vec{r}$ 为数组引用 R_A 在循环中的重用向量,$ML_{R_A}(\vec{i})$ 和 $ML_{R_B}(\vec{j})$ 分别为数组引用 R_A 和 R_B 在迭代点 $\vec{i}$ 和 $\vec{j}$ 上访问数据的内存地址,n 和 b 为构造 CME 过程中引入的人工变量。

假定循环的层数为 m,对于抽样过程中确定的一个迭代点 $\vec{i} = \vec{i}_0 = (i_{10}, i_{20}, \cdots, i_{m0})$,$ML_{R_A}(\vec{i}_0)$ 和 L_{off} 的值可以得到确定,而 b 的范围也因为 L_{off} 的确定而得到确定,因此上述冲突 CME 可以转化为如下形式:

$$\begin{cases} ML_{R_B}(\vec{j}) + n * C + b = W \\ -L_{off} < b \leqslant L - 1 - L_{off}, L_{off} = ML_{R_A}(\vec{i}_0) \bmod L, n \neq 0 \\ \vec{j} \in [\vec{p}_0, \vec{i}_0), \vec{p}_0 = \vec{i}_0 - \vec{r} \\ W = ML_{R_A}(\vec{i}_0), C = C_s/k \end{cases}$$

若数组 B 为 N 维数组,且各维大小为 $M_i (i = 1, 2, \cdots N)$,其数组引用具有如下形式:

$$R_B = B(f_1(\vec{j}), f_2(\vec{j}), \cdots f_N(\vec{j}))$$

则 $ML_{R_B}(\vec{j}) = B$ 数组内存首地址 $+ \sum_{i=1}^{N} f_i(\vec{j}) \prod_{k=0}^{N-1} M_k$

由于循环中数组引用的下标表达式为线性表达式,即 $f_k(\vec{j})(1 \leqslant k \leqslant N)$ 为迭代 $\vec{j}$ 的仿射函数,因此 CME 中的等式经过转换可以表示成为如下形式:

$$a_1 j_1 + a_2 j_2 + \cdots + a_m j_m + n * C + b = W' \quad (1)$$

其中 $\vec{j} = (j_1, j_2, \cdots, j_m) \in [\vec{p}_0, \vec{i}_0)$。

若方程(1)存在整数解 $\vec{j} \in [\vec{p}_0, \vec{i}_0)$,即表明数组引用 R_A 在迭代点 $\vec{i}_0$ 上沿着重用向量 $\vec{r}$ 的数据访问是 Cache 不命中的,否则是 Cache 命中的。

*)国家自然科学基金(100720077)。舒 辉 博士生,研究方向:并行编译。

对于约束条件 $\vec{j}\in[\vec{p}_0,\vec{t}_0)$，若重用向量 $\vec{r}=(r_1,\cdots,r_{k-1},r_k,\cdots,r_m)$ 的前 k－1个分量为0($r_i=0,1\leqslant i\leqslant k-1$)，那么 $(j_1,\cdots,j_{k-1})=(i_{10},\cdots,i_{k-1,0})$，而 $(j_k,\cdots,j_m)$ 的取值集合则可以表示成2$(m-k)$＋1个凸集的并集[1]。

对于数组引用的重用发生在循环的内层和次内层，丢番图方程中的变量数目(人工变量除外)不超过两个，而 n 和 b 的取值范围是有限的，对此可以通过通用 Banerjee 算法求出方程整数解的通解[6]，带入约束条件中去测试约束条件是否仍然成立即可，对应的时间复杂度不会很高，对此本文不作过多的讨论。但当数组引用的重用发生在循环的第三层(由内向外)或更高层时，变量的数组往往超过了3个，此时若采用通用 Banerjee 算法去进行测试则需要遍历一个或多个变量组合的可能取值，显然这是不合适的，我们在下一节给出格测试的方法，试图更好地解决这个问题。

3. 格测试

我们知道，对于一个没有约束条件的丢番图方程而言，只要方程变量系数的最大公约数能够整除方程的常数项，那么该方程就必然含有整数解。用于表示该方程整数解的一种形式是用该方程的一个特解加上该方程对应齐次方程解基的线性组合。这样的表示形式构成了丢番图方程整数解的解空间。当我们给出丢番图方程的约束条件时，即是要判断该解空间与约束条件构成的空间在整数域上是否存在交集。在无法直接判断的情况下，只能通过用基的线性组合加上特解去遍历约束条件空间的方法来进行测试。我们假定齐次方程的一组基向量为$(\vec{b}_1,\cdots,\vec{b}_n)$，若基向量之间是近似正交的，那么 $H+k\vec{b}_n$ 和 $H+(k+1)\vec{b}_n(k\in Z)$ 两个整数域上连续的超平面之间的距离就有可能比较长(H 是由 $\vec{b}_1,\cdots,\vec{b}_{n-1}$ 构成的子空间)，那么 k 满足约束条件的取值范围就会变小，换言之，对空间遍历的测试次数就会减少。

美国 Utrecht 大学的 Karen 博士给出了一个完整的求丢番图方程整数解通解的数学方法[5]，他根据给定的丢番图方程构造一个格空间的基向量矩阵 B，对 B 实施 L^3 格约化算法后得到新的基向量矩阵 B'，B' 中的基向量(格约基)是近似正交的，具体地有：

对于方程 $\sum_{i=1}^{n}a_ix_i=d$，其中 GCD$(a_1,\cdots,a_n)=1$。令 $\vec{a}=(a_1,\cdots,a_n)$，$\vec{x}=(x_1,\cdots,x_n)^T$，构造一个格，其基向量的形式如下：

$$(x_1,\cdots,x_n,N_1y,N_2(\vec{a}\vec{x}-dy))^T$$

其中 N_1，N_2 是整数，y 为自由变量。

对于该形式的格，一个基本的基向量矩阵形式如下：

$$B=\begin{pmatrix} I^{(n)} & 0^{(n*1)} \\ 0^{(1*n)} & N_1 \\ N_2\vec{a} & -N_2d \end{pmatrix}$$

Karen 证明只要 N_1，N_2 取得足够大，对 B 矩阵实施格约化算法之后，得到的新矩阵 B' 的前 n 列将有如下形式：

$$\begin{pmatrix} C_{n*n-1} & \vec{x}_d \\ 0^{(1*(n-1))} & N_1 \\ 0^{(1*(n-1))} & 0 \end{pmatrix}$$

对于上式，C 矩阵中列向量构成了 $\sum_{i=1}^{n}a_ix_i=0$ 方程的一组近似正交基，x_d 是 $\sum_{i=1}^{n}a_ix_i=d$ 的一个特解。通过 Karen 的方法，我们可以得到方程整数通解的一个表示形式：

$$\vec{x}=\vec{x}_d+C_{n*n-1}\begin{pmatrix}\lambda_1\\ \lambda_2\\ \vdots\\ \lambda_{n-1}\end{pmatrix}$$

有了这个通解形式，我们就可以利用它对丢番图方程的约束条件空间进行遍历。

若丢番图方程的约束条件的形式为$0\leqslant\vec{x}\leqslant U$，将上述通解形式带入并经过整理可以表示成如下形式：

$$D_{(2n*(n-1))}\begin{pmatrix}\lambda_1\\ \lambda_2\\ \vdots\\ \lambda_{n-1}\end{pmatrix}\leqslant\begin{pmatrix}e_1\\ e_2\\ \vdots\\ e_{n-1}\end{pmatrix}$$

选定一个变量 λ_k，依据 d_{ik} 的符号将上述不等式分成三组，每一组不等式得基本形式为：

①$\sum_{j=1}^{2n}d_{ik}<e_i$，当 $d_{ik}=0$时；

②$d_{ik}\lambda_k\leqslant e_i-\sum_{j=1}^{k-1}d_{ij}\lambda_j-\sum_{j=k+1}^{2n}d_{ij}\lambda_j$，当 $d_{ik}>0$时；

③$-e_i+\sum_{j=1}^{k-1}d_{ij}\lambda_j+\sum_{j=k+1}^{2n}d_{ij}\lambda_j<d_{ik}\lambda_k$，当 $d_{ik}<0$时。

通过分组，我们就可以得到 λ_k 的上界和下界，对于可以进行 Ceiling and Flooring 的不等式，实施变量系数的约化，对不等式组进行傅立叶-密西根变量削减(即将②和③中关于 λ_k 的上界和下界两两组合，比如对于 $L\leqslant c_1\lambda_k$ 和 $c_2\lambda_k\leqslant U$，$c_1,c_2>0$，可以得到不等式 $c_2L\leqslant c_1U$，该不等式中不包含 λ_k)，削去变量 λ_k 得到新的不等式组，如此递归削减直到不等式组只剩一个变量，就可以得到变量的最小值和最大值。若最小值大于最大值，那么显然方程无解，否则将变量可能的取值逐一代入原不等式组中，对新不等式组重新进行傅立叶-密西根变量削减，如此递归遍历下去，直到完成对丢番图方程有无整数解的判定。

遍历方向的选择：我们选择齐次方程解基中欧几里德长度最长的变量作为削减应该保留的变量，这是基于一种如图1所示的认识。

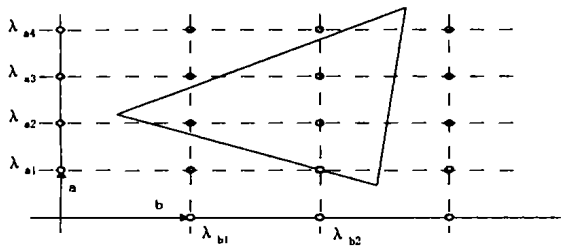

图1

图1中的三角形表示丢番图方程的约束空间，图中的阴影点表示丢番图方程的整数解，由基向量 a 和 b 的整数线性组合可以表示所有的整数解，从图中可以看出，若我们沿着 b 方向(长度较长)遍历，对应的 λ_b 的取值只有两个，而我们沿着 a 方向遍历，对应的 λ_a 的可能取值就为四个，显然我们从欧几里德长度较长的方向遍历比较合适。当然，实际上在不同方向上的遍历次数可能和约束空间的具体形状有关，所谓从欧几里德长度较长的变量开始遍历，只是一种遍历方向上的选择，我们只是这样认为在这种遍历方式下，可能的遍历次数会较少。

遍历值的选择：在求出了变量的最小和最大值之后，并不机械地从变量的最小值开始遍历，而是从变量的取值范围中

选择合适的值去进行遍历，因为通过 L^3 格约化算法求出的丢番图方程的特解的欧几里德长度较短，我们认为该特解只是在约束条件构成的边界附近，若该特解不满足丢番图方程的约束条件，对特解向量中不满足约束条件的分量，我们选择合适的变量值，使该分量能够满足约束条件，那么我们沿着这个方向深度递归遍历下去，对于丢番图方程有整数解的情况而言，就有可能迅速判定方程整数解的存在性。

经过上述分析之后，我们给出示意性的格测试算法：

算法输入：带线性约束条件的多元一次丢番图方程。

算法输出：丢番图方程整数解的存在性。

算法步骤：

步骤1　丢番图方程线性约束条件的正规化。

步骤2　依据 karen 的算法依据丢番图方程构造对应格的基本基并根据 L^3 格约化算法求出格约基。

步骤3　若方程特解已经满足给定的约束条件，则方程有解，算法结束。

步骤4　依据格约基得到方程通解的形式，重新构造约束条件。

步骤5　对约束条件不等式进行 ceiling 和 flooring。

步骤6　选择丢番图方程对应齐次方程解基中欧几里德长度较长的基向量对应的变量作为傅立叶-密西根变量削减应该保留的变量，对约束条件不等式组进行傅立叶-密西根变量削减，求出保留变量的最小值和最大值，若最小值大于最大值，丢番图方程无解，算法结束。

步骤7　在该变量的最小值和最大值之间选择合适的值，对变量赋值，重新构造约束条件，对约束条件空间进行深度递归遍历，直至判定方程有解或无解。

4. 格测试算法实例

本节我们给出一个运用格测试算法对 FORTRAN 程序循环段（来自 NASA 7 kernels BenchMark 的 BTRIX）进行 CME 抽样分析的实例：

```
real S(60,60,60,5),B(5,5,60,60)
DO 200 J = 2,58
   DO 200 M = 1,5
      DO 200 L = 2,59
         DO 200 K = 1,5
            S(J,1,L,M)= S(J,1,L,M) * B(M,K,J,L) * S(J
            +1,1,L,K)
CONTINUE
```

循环段中的数组引用 S(J,1,L,M)具有空间重用向量(1,0,0,0)。我们用随机抽样的方法来考察数组引用 S(J,1,L,M)和 B(M,K,J,L)之间在其重用向量上的 Cache 冲突情况（直接映射 Cache 容量为32k，行长为32）。随机抽取一个迭代点(4,2,24,3)，构造相应的 CME 方程，经过约束条件的正规化（过程略），最终形成如下的丢番图方程：

$32768n+1500l+5k+m=57973$

约束条件为：$0\leqslant n\leqslant 2, 0\leqslant l\leqslant 57, 0\leqslant k\leqslant 4, 0\leqslant m\leqslant 33$

首先构造格的一个基本基：

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1000 \\ 3.2768E8 & 1.5E7 & 5E4 & 1E4 & -5.7973E8 \end{pmatrix}$$

对该基进行格约化得到格约基如下（格约化时间在 PIII-800PC 机上经过实测<0.001s）：

$$\begin{pmatrix} 0 & 1 & -5 & 2 & 0 \\ 0 & -22 & 109 & -5 & 0 \\ -1 & 45 & 65 & -12 & 0 \\ 5 & 7 & 15 & -3 & 1 \\ 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 10000 \end{pmatrix}$$

在该组基中，可以验证格约基矩阵的前三列基向量的前4个元素构成的分向量(0,0,-1,5)、(1,-22,45,7)、(-5,109,65,15)为方程 $32768n+1500l+5k+m=0$ 解空间的一组近似正交基，矩阵中第4列向量的前4个元素构成的分向量(2,-5,-12,-3)为原丢番图方程的一个特解，该特解的欧几里德长度较短。

在此基础上，我们有了一个该丢番图方程通解的一个特殊形式：

$$\begin{pmatrix} n \\ l \\ k \\ m \end{pmatrix} = \begin{pmatrix} 2 \\ -5 \\ -12 \\ -3 \end{pmatrix} + \begin{pmatrix} 0 & 1 & -5 \\ 0 & -22 & 109 \\ -1 & 45 & 65 \\ 5 & 7 & 15 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix}$$

重新构造丢番图方程的约束空间：

$$\begin{cases} 0\leqslant 2+\lambda_2-5\lambda_3\leqslant 2 \\ 0\leqslant -5-22\lambda_2+109\lambda_3\leqslant 57 \\ 0\leqslant -12-\lambda_1+45\lambda_2+65\lambda_3\leqslant 4 \\ 0\leqslant -3+5\lambda_1+7\lambda_2+15\lambda_3\leqslant 33 \end{cases}$$

对这样一个由 $(\lambda_1,\lambda_2,\lambda_3)$ 构成的不等式组，运用傅立叶-密西根变量削减算法进行变量削减，我们通过削去变量 λ_1 和 λ_2 可以得出变量 λ_3 的一个约束条件：

$1035\leqslant 35880\lambda_3\leqslant 13312$

显然，λ_3 没有整数解，也即 $(\lambda_1,\lambda_2,\lambda_3)$ 没有整数解，因而有丢番图方程在其约束条件内没有整数解。

我们对这个程序实例中全部迭代点进行 CME 分析，该循环段迭代总数为826509个迭代点，由于数组引用 S(J,1,LM)具有长空间重用向量(1,0,0,0)，其与 B(M,K,J,L)在该重用向量上的冲突 CME 中的丢番图方程为多变量丢番图方程，在实际分析中需要用到格测试的丢番图方程总数为14500个，按格测试算法对约束空间递归遍历的节点数统计如表1所示，所有这些格测试的时间总和经统计为29秒（PIII-800 PC 机），对单个迭代点的平均测试时间为0.002秒。在 CME 抽样分析中，对该循环的抽样迭代点数为862个，总的分析时间为1.1秒。

表1

递归遍历节点数	0～10	11～20	21～30	>31
方程个数	12479	867	921	233

参 考 文 献

1 Ghosh S, Martonosi M, Malik S. Cache Miss Equations: A compiler framework for analyzing and tuning memory behavior. In ACM transactions on Programming Languages and Systems. 1999

2 Dantzig G B, tis Eabes B C. Fourier-Motzkin elimination and its dual. Journal of Combinatorial Theory, 1973, 14: 288～297

3 Wolfe MJ, Tseng CW. The power test for data dependence: [Technical Report CSE 90-015]. Oregon Graduate Institute, 1990

4 Cohen H. A Course in Coputational Algebraic Number Theory. 世界图书出版公司, 1997. 84～95

5 Aardal K, Hurkens C A J, Lenstra A K. Solving a system of diophantine equation with lower and upper bounds on the variables. Mathematics of Operations Research, 2000, 25: 427～442

6 Banerjee U. Dependence analysis for supercomputing. Kluwer Academic Publishers, 1988