Р.Н. Арапбаев, Р.А. Осмонов

АНАЛИЗ ЗАВИСИМОСТЕЙ ПО ДАННЫМ ДЛЯ МНОГОМЕРНЫХ МАССИВОВ НА БАЗЕ МОДИФИЦИРОВАННОГО λ-ТЕСТА

ВВЕДЕНИЕ

Одним из основных средств перевода последовательных программ в параллельную форму являются автоматические распараллеливающие компиляторы. Основная их задача — извлечь как можно больше скрытого параллелизма из последовательной программы. Главным источником такого потенциального параллелизма, как правило, служит гнездо цикла. Извлечение скрытого параллелизма в первую очередь связано с анализом циклов и заключается в нахождении зависимости по данным между итерациями цикла.

Для решения этой проблемы компиляторы используют тесты на зависимость по данным [1]. На практике используются несколько известных алгоритмов анализа зависимости по данным. Например, НОД-тест, неравенства Банержи [2], I-тест (интервальный тест) [3, 4], Power-тест [5], Омега-тест [6], λ-тест [7] и др.

В настоящей работе предлагается новый модифицированный вариант λ-теста. В приведенном алгоритме λ-тест интегрирован с точным IR-тестом (“interval reduction”) [8], благодаря чему он показал более точные результаты при анализе зависимостей многомерных массивов.

В разд. 2 даны основные определения, в разд. 3 представлены идеи λ-теста и IR-теста. В разд. 4 подробно описывается алгоритм модифицированного λ-теста и производится сравнение его результатов с результатами наиболее известных тестов на зависимость. В разд. 5 приводится заключение о проделанной работе и список литературы по данной тематике.

Все понятия, не определяемые в этой работе, могут быть найдены в [1].

1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Рассмотрим гнездо циклов с индексами $i_1, i_2,\ldots, i_n$:

```
for   i1 = L1 to U1
  for   i2 = L2 to U2
                 ⋮
      for   ir = Lr to Ur
S1:       A[f1(i1, i2, … , ir), f2(i1, i2, . . . , ir), …, fd(i1, i2,…, ir)] = . . .
S2:          ...=A[f′1(i1, i2,..., ir), f′2(i1, i2,…, ir), …, f′d(i1, i2,…, ir)]
      endfor
                 ⋮
   endfor
endfor
```

где **d** — размерность массива, **r** — количество вложенных циклов.

Определение 1. Между двумя операторами $\mathbf{S_1}$ и $\mathbf{S_2}$ существует зависимость, если они оба обращаются к одной и той же ячейке памяти, и, по крайней мере, одно из этих обращений есть *запись* [1].

Компиляторы применяют тесты на зависимость, чтобы определить, обращаются ли операторы $\mathbf{S_1}$ и $\mathbf{S_2}$ к одному и тому же элементу массива **A**. Для этого тест должен учитывать не только индексные выражения $f_1(),$ $f_2(),\ldots,f_d(),f'_1(),$ $f'_2()\ldots,$ $f'_d(),$ но также границы индексных переменных $L_1,L_2\ldots,L_n,U_1,U_2\ldots,U_n$. Таким образом, тесты на зависимость должны определять, есть ли целочисленные решения $i_1,i_2\ldots,$ i_n системы линейных диофантовых уравнений (1), удовлетворяющие ограничениям (2):

$$\begin{cases} f_1(i_1,i_2,\ldots,i_r) = f'_1(i_{r+1},i_{r+2},\ldots,i_n) \\ f_2(i_1,i_2,\ldots,i_r) = f'_2(i_{r+1},i_{r+2},\ldots,i_n) \\ \quad\vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ f_d(i_1,i_2,\ldots,i_r) = f'_d(i_{r+1},i_{r+2},\ldots,i_n) \end{cases} \tag{1.1}$$

$$L_p \le i_p \le U_p, \text{ где } p=1,\ldots,n. \tag{1.2}$$

Ключевой проблемой при анализе зависимостей в цикле является работа с массивами. Если имеется m-мерный массив **A** и индексные выражения массива линейны, то тогда система (1.1) может быть записана в следующем виде:

$$
\begin{array}{l}
a_{11} x_1 + a_{12} x_2 + \ldots + a_{1\mathrm{n}} x_\mathrm{n} + c_1 = 0 \\
a_{21} x_1 + a_{22} x_2 + \ldots + a_{2\mathrm{n}} x_\mathrm{n} + c_2 = 0 \\
\qquad\qquad \ldots\ldots\ldots\ldots \\
a_{m1} x_1 + a_{m2} x_2 + \ldots + a_{m\mathrm{n}} x_\mathrm{n} + c_m = 0
\end{array}
\tag{1.3}
$$

и

$$L_i \le x_i \le U_i \text{ где } i = 1,\ldots,\mathrm{n} \tag{1.4}$$

Следовательно, проблема зависимости представляет собой задачу целочисленного программирования.

Общий подход состоит в индивидуальном тестировании уравнений (тестирование “индекс-за-индексом”) из (1.3) вместо проверки существования решения системы в целом. Но система уравнений зависимости может не иметь решения даже в том случае, когда имеются решения в каждом из отдельных уравнений.

Определение 2. Будем говорить, что индексные выражения *сцепленные* (“coupled”), если они включают в себя одинаковые индексные переменные цикла.

Если потенциальная зависимость включает *сцепленные* индексы, то для её разрушения необходимо одновременное рассмотрение индексов многомерного массива. Заметим, что среди всего множества тестов только некоторые из них пригодны для работы со *сцепленными* индексами. Например, обобщенный НОД-тест и тесты на основе линейного и целочисленного программирования: целочисленной тест, Power-тест, Омега-тест, и др. В первом даётся ответ на существование целочисленного решения, но не учитываются ограничения на область изменения переменных. Поэтому обобщенный НОД-тест не может доказать существование зависимости, но полезен для ее опровержения. Тесты, использующие дорогостоящие методы, неэффективны на практике. Экспериментальные результаты показали, что метод исключения переменных Фурье—Моцкина выполняется в 22-28 раз дольше, чем при использовании более простых методов [1].

Согласно эмпирическому исследованию в [9], сцепленные индексы часто встречаются в реальных программах. Из всех исследованных массивов 36.23% составляют ссылки двухмерных массивов и 7% — ссылки трехмерных массивов. Доля ссылок выше трехмерного массива незначительна. Более чем в четырех тысячах пар двухмерных ссылок массива приблизительно 46% являются сцепленными индексными выражениями. Что касается ссылок массива большей размерности, то только 2% являются сцепленными индексными выражениями. Поэтому на практике важно иметь эффективный тест для обработки сцепленных индексов, особенно для анализа ссылок двухмерного массива. Один из таких эффективных алгоритмов, называемый λ-тестом, предложен в работе [7].

2. ИСПОЛЬЗУЕМЫЕ ТЕСТЫ

2.1. λ-тест

λ-тест исследует систему уравнений (1.3) и неравенства (1.4) и определяет, имеет ли система действительные решения [7].

Геометрически каждое линейное уравнение в (1.3) представляет собой гиперплоскость π в пространстве $\mathbf{R^n}$. Пересечение **m** гиперплоскостей ***S*** соответствует общим решениям системы (1.3). Очевидно, если ***S*** пусто, то не имеется никакой зависимости по данным. Границы циклов соответствуют ограниченному выпуклому множеству **V** в $\mathbf{R^n}$. Уравнение имеет действительное решение, удовлетворяющее границам циклов и направлениям зависимостей, тогда и только тогда, когда соответствующая уравнению гиперплоскость π пересекается с **V**. Тестирование «индекс-за-индексом» определяет, пересекается ли каждая гиперплоскость π с **V**. Необходимо определить, пересекается ли само **S** с **V**. Если из всех гиперплоскостей найдется такая гиперплоскость, которая не пересекает **V**, то очевидно **S** не может пересекаться с **V**. Однако, даже если каждая гиперплоскость из (1.3) пересекает **V**, существует вероятность, что **S** не пересечет **V**.

На рис. 1 π_1 и π_2 — гиперплоскости, представляющие два уравнения системы, каждая из которых пересекается с **V**. Предположим, что пересечение π_1 и π_2 находится вне **V**. Если можно найти новую гиперплоскость, которая содержит **S**, но не пересекает **V**, то это доказывает, что **S** и **V** не пересекаются. На рис. 1 π' является такой новой гиперплоскостью. Следующая теорема доказывает, что если **S** и **V** не имеют пересечения, то имеет место гиперплоскость в $\mathbf{R^n}$, которая содержит **S** и не пересекает **V**. Данная гиперплоскость является линейной комбинацией гиперплоскостей из (1.3). С другой стороны, если **S** и **V** пересекутся, то никакая такая линейная комбинация не существует.

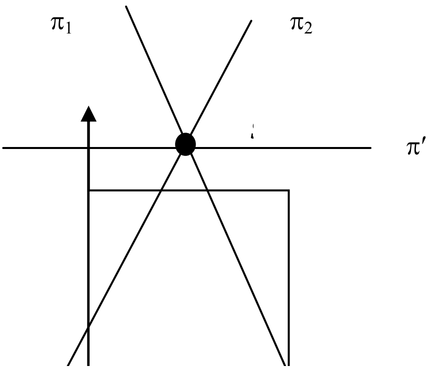

Рис. 1. Геометрическая иллюстрация λ-теста

Теорема 1. S∩V = ∅ тогда и только тогда, когда существует гиперплоскость π, определяемая линейной комбинацией уравнений из (1.3): $\left\langle \sum_{i=1}^{m} \lambda_i \vec{a}_i, \vec{x} \right\rangle + \sum_{i=1}^{m} \lambda_i c_i = 0$ такая, что π ∩V = ∅, где $<\vec{a}_i, \vec{x}>$ — скалярное произведение векторов $\vec{a}_i \equiv (a_{i1}\ a_{i2}, \ldots, a_{in})$ и $\vec{x} \equiv (x_1, x_2, \ldots x_n)$ [7].

Массив (λ_1, λ_2, ..., λ_m) в Теореме 1 определяет гиперплоскость, содержащую **S**. Имеется бесконечное число таких гиперплоскостей. Задача λ-теста — исследовать по мере необходимости некоторое количество гиперплоскостей для определения пересечения **S** и **V**. Согласно Теореме 3 из [7] в общем случае, λ-тест генерирует C_n^{m-1} таких гиперплоскостей, которые являются линейной комбинацией (1.3) и называются λ-плоскостями. Чтобы определить, пересекает ли каждая λ-плоскость **V**, применяется тест Банержи—Вульфа для каждой λ-плоскости. Если хотя бы одна из λ-плоскостей не пересекает **V**, тогда нет зависимости по данным. Если каждая λ-плоскость пересекает **V**, то λ-тест принимает решение о возможном существовании зависимости.

2.2. IR-тест

IR-тест находит целочисленные решения уравнения зависимости путем сокращения интервала решений переменных с многократным проектированием. Как только эффективный интервал решений какой-нибудь переменной сжимается к пустому, то это линейное диофантово уравнение не имеет целочисленного решения [8].

Для объяснения IR-теста введем некоторые понятия.

Пусть, a — целое число, тогда положительная часть числа a:
$a^+ = \max\{a,0\}$, отрицательная часть числа a: $a^- = \max\{-a,0\}$.

Пусть L,U — числа такие, что L≤U, тогда
$\min\{ax : L \le x \le U\} = a^+ L - a^- U$,
$\max\{ax : L \le x \le U\} = a^+ U - a^- L$.

Пусть $n \ge 1$, a_j, L_j, U_j — числа и $L_j \le U_j$ для всех $j \in [1:n]$, и $\sum_{1\le j\le n} a_j x_j$ – линейная функция n переменных, тогда:

$$\min\left\{\sum_{1\le j\le n} a_j x_j : (x_1,...,x_n) \in \bigtimes_{1\le j\le n} [L_j : U_j]\right\} = \sum_{1\le j\le n} (a_j^+ L_j - a_j^- U_j),$$

$$\max\left\{\sum_{1\le j\le n} a_j x_j : (x_1,...,x_n) \in \bigtimes_{1\le j\le n} [L_j : U_j]\right\} = \sum_{1\le j\le n} (a_j^+ U_j - a_j^- L_j).$$

Поскольку линейную функцию двух переменных можно представить как линию в двухмерном пространстве, то говорят, что линия $ax + by + c = 0$ содержит в себе *целую точку* тогда и только тогда, когда существует целочисленная пара (x_1, y_1) такая, что $ax_1 + by_1 + c = 0$. В связи с этим определением имеем следующую лемму.

Лемма 1. Пусть $P_1(\lfloor x_1 \rfloor, y_1)$ и $P_2(\lceil x_1 \rceil, y_2)$ будут двумя точками на линии $ax + by + c = 0$, где x_1 — произвольное вещественное число, $\lfloor x_1 \rfloor$ — нижняя целая часть числа x_1, и $\lceil x_1 \rceil$ — верхняя целая часть числа x_1. Тогда отрезок $\overline{P_1P_2}$ не содержит целую точку тогда и только тогда, когда y_1 и y_2 являются нецелыми (рис. 2).

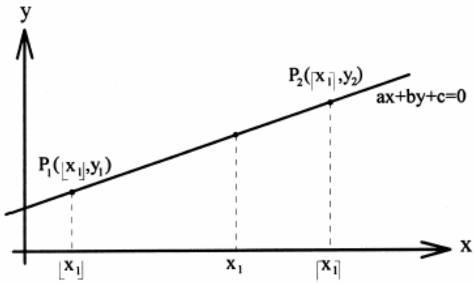

Рис. 2. Линейное уравнение в двухмерном пространстве

Следующая теорема является прямым обобщением вышеупомянутой леммы.

Теорема 2. Пусть $P_1(\lfloor x_1\rfloor, x_2, \ldots, x_n)$ и $P_2(\lceil x_1\rceil, x'_2, \ldots, x'_n)$ две точки на линии $\sum_{1\le j\le n} a_j x_j + c = 0$ в n-мерном пространстве. Тогда отрезок $\overline{P_1P_2}$ не содержит целую точку тогда и только тогда, когда

(i) $\exists k \in [2{:}n]$ такое, что $x_k \notin Z$ и

(ii) $\exists m \in [2{:}n]$ такое, что $x'_m \notin Z$.

Чтобы более ясно показать идею IR-теста, сначала рассматривается случай, когда уравнение зависимости представляет собой диафонтово уравнение с двумя переменными.

Процедура теста в случае уравнения с двумя переменными приведена ниже.

$$a_1x_1 + a_2x_2 = a_0 \qquad (2.2.1)$$

при условии

$$l_1 \le x_1 \le u_1 \text{ и } l_2 \le x_2 \le u_2, \qquad (2.2.2)$$

где a_0, a_1, a_2 являются целыми константами, и x_1, x_2 — целочисленные переменные. Очевидно, что целочисленные решения будут размешаться внутри ограниченной прямоугольной области: $l_1 \le x_1 \le u_1$ и $l_2 \le x_2 \le u_2$.

Чтобы найти целочисленные решения, линия проектируется на ось x_1. Эффективным интервалом решений x_1 является пересечение интервала линии, спроектированной на ось x_1, с исходным интервалом $l_1 \le x_1 \le u_1$. Таким образом, фактическим интервалом решений x_1 будет подмножество начального интервала ограничения.

Проектируемым интервалом линии $a_1x_1 + a_2x_2 = a_0$ на ось x_1 является:

$x_1 = -(a_2/a_1)\, x_2 + (a_0/a_1)$, где $l_2 \le x_2 \le u_2$.

Вычисляется предельное значение x_1:

$$\left(-\frac{a_2}{a_1}\right)^+ l_2 - \left(-\frac{a_2}{a_1}\right)^- u_2 + \frac{a_0}{a_1} \le x_1 \le \left(-\frac{a_2}{a_1}\right)^+ u_2 - \left(-\frac{a_2}{a_1}\right)^- l_2 + \frac{a_0}{a_1}.$$

Пусть

$$l_1^{(1)} = \left(-\frac{a_2}{a_1}\right)^+ l_2 - \left(-\frac{a_2}{a_1}\right)^- u_2 + \frac{a_0}{a_1},$$

$$u_1^{(1)} = \left(-\frac{a_2}{a_1}\right)^+ u_2 - \left(-\frac{a_2}{a_1}\right)^- l_2 + \frac{a_0}{a_1}$$

означает верхнюю и нижнюю границы x'_1 соответственно.

Фактический интервал решений x_1 :

$$\left[l_1^{(1)} : u_1^{(1)}\right] \cap \left[l_1 : u_1\right] = \left[l_1^{'(1)} : u_1^{'(1)}\right].$$

Так как $l_1^{'(1)}$ и $u_1^{'(1)}$ могут быть нецелочисленными по Лемме 1, и если уравнение (2.2.1) имеет целочисленные решения, то они должны принадлежать интервалу [$\lceil l_1^{'(1)} \rceil : \lfloor u_1^{'(1)} \rfloor$]. Если $\lceil l_1^{'(1)} \rceil > \lfloor u_1^{'(1)} \rfloor$, то интервал пуст и уравнение (2.2.1) с ограничениями (2.2.2) не имеет целочисленных решений. Иначе, так как интервал решений x_1 был сокращен (рис. 3), интервал решений x_2 изменится соответственно.

Аналогично, как описано выше, линия проектируется на ось x_2. Пусть сокращенный интервал решений x_2 будет [$\lceil l_2^{'(1)} \rceil : \lfloor u_2^{'(1)} \rfloor$] (рис. 3). Если теперь $\lceil l_2^{'(1)} \rceil > \lfloor u_2^{'(1)} \rfloor$, то уравнение (2.2.1.) с ограничениями (2.2.2) не имеет целочисленных решений. Иначе линия снова проектируется на эту ось x_1 с целью сокращения интервала решений x_1.

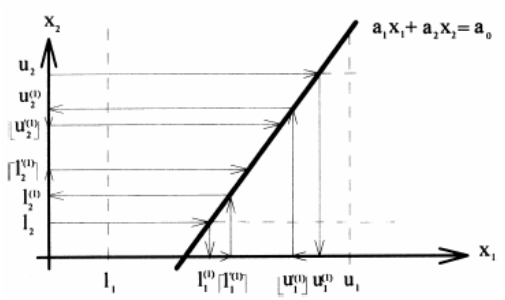

Рис. 3. Сокращение интервала решения

Повторив эту процедуру *m* раз, имеем:

$x_1 \in [\lceil l_1^{'(m)} \rceil : \lfloor u_1^{'(m)} \rfloor],$
$x_2 \in [\lceil l_2^{'(m)} \rceil : \lfloor u_2^{'(m)} \rfloor].$

Повторяя $(m + 1)$ раз, приходим к

$x_1 \in [\lceil l_1^{'(m+1)} \rceil : \lfloor u_1^{'(m+1)} \rfloor],$
$x_2 \in [\lceil l_2^{'(m+1)} \rceil : \lfloor u_2^{'(m+1)} \rfloor].$

В процедуре нахождения целочисленного решения уравнения (2.2.1) при условии ограничения (2.2.2), имеются следующие случаи, где IR-тест принимает соответствующие решения.

1. Если интервалы решений x_1 или x_2 были сокращены к пустым, т. е. $\lceil l_1^{'(m)} \rceil > \lfloor u_1^{'(m)} \rfloor$ или $\lceil l_2^{'(m)} \rceil > \lfloor u_2^{'(m)} \rfloor$, то уравнение (2.2.1) с ограничением (2.2.2) не имеет целочисленных решений, IR-тест показывает независимость по данным.
2. Если интервалы решений x_1 и x_2 остаются неизменными, т.е. $\lceil l_1^{'(m)} \rceil = \lceil l_1^{'(m+1)} \rceil, \lfloor u_1^{'(m)} \rfloor = \lfloor u_1^{'(m+1)} \rfloor$ и $\lceil l_2^{'(m)} \rceil = \lceil l_2^{'(m+1)} \rceil, \lfloor u_2^{'(m)} \rfloor = \lfloor u_2^{'(m+1)} \rfloor$, то уравнение (2.2.1) с ограничением (2.2.2) содержит, по крайней мере, одно целочисленное решение.

(a) Если оба интервала решений сократились к одной точке, т.е. $\lceil l_1^{'(m)} \rceil = \lfloor u_1^{'(m)} \rfloor = \lceil l_1^{'(m+1)} \rceil = \lfloor u_1^{'(m+1)} \rfloor$ и $\lceil l_2^{'(m)} \rceil = \lfloor u_2^{'(m)} \rfloor = \lceil l_2^{'(m+1)} \rceil = \lfloor u_2^{'(m+1)} \rfloor$,
то уравнение (2.2.1) с ограничением (2.2.2) имеет только одно целочисленное решение: $(x_1,x_2) = (\lceil l_1^{'(m)} \rceil, \lceil l_2^{'(m)} \rceil)$.

(b) Иначе, уравнение (2.2.1) с ограничением (2.2.2) содержит, по крайней мере, два целочисленных решения:
$\{(\lceil l_1^{'(m)} \rceil, \lceil l_2^{'(m)} \rceil), (\lfloor u_1^{'(m)} \rfloor, \lfloor u_2^{'(m)} \rfloor)\}$ если $(-a_1/a_2)>0$,
$\{(\lceil l_1^{'(m)} \rceil, \lfloor u_2^{'(m)} \rfloor), (\lfloor u_1^{'(m)} \rfloor, \lceil l_2^{'(m)} \rceil)\}$ если $(-a_1/a_2)<0$

Для нахождения других целочисленных решений в случае (2b), интервалы решений x_1 и x_2 уменьшаются, добавляя и вычитая единицу к нижней и верхней границам соответственно. Данные интервалы используются в качестве новых интервалов, и вышеупомянутая процедура повторяется до тех пор, пока один из интервалов не станет пустым. В конечном счете, все целочисленные решения могут быть найдены.

В случае уравнения, содержащего n переменных, тест решает уравнение зависимости, рекурсивно применяя для каждой переменной приведенный метод [8].

3. МОДИФИЦИРОВАННЫЙ λ–ТЕСТ

В этом разделе представлен алгоритм модифицированного λ-теста. Рассматривается проблема зависимости по данным при условии, что индексные выражения массивов линейны. Границы циклов рассматриваются как постоянные.

В модифицированном λ-тесте вместо Банержи-теста используется более мощный IR-тест. Поясним алгоритм при помощи геометрической иллюстрации. Геометрически каждое линейное уравнение в (1.3) представляет собой гиперплоскость π в пространстве $\mathbf{R^n}$. Пересечение **m** гиперплоскостей — ***S*** соответствует общим решениям всех уравнений (1.3). Очевидно, если ***S*** пусто, то не существует никакой зависимости по данным. Проверка, является ли ***S*** пустой, тривиальна в линейной алгебре. Поэтому далее рассматриваем только непустую ***S***. Границы циклов соответствуют ограниченному выпуклому множеству **V** в $\mathbf{R^n}$. Система уравнений (1.3) имеет действительное решение, удовлетворяющее границам циклов тогда и только тогда, когда ***S*** пересекается с **V**. λ-тест может ответить на этот вопрос.

Обычный λ-тест доказывает независимость по данным многомерных массивов, когда **S** не пресекает выпуклое множество **V,** в противном случае λ-тест принимает консервативное решение о существовании зависимости по данным. Банержи-тест не может определить, имеет ли λ-плоскость целочисленные точки пересечения с **V.**

Однако, даже если ***S*** пересекается с **V**, существует вероятность, что система уравнений (1.3) не имеет целочисленных решений в **V**.

Если можно найти новую гиперплоскость, которая содержит ***S*** и не имеет целочисленных точек пересечения с **V**, то это доказывает, что ***S*** не имеет целочисленных значений в **V**, следовательно, зависимости по данным не существует. Данная гиперплоскость является линейной комбинацией уравнений системы.

Теорема 3. **S**$\cap$**V** не имеет целочисленных точек тогда и только тогда, когда существует гиперплоскость π, соответствующая линейной комбинации $\left\langle \sum_{i=1}^{m} \lambda_i \vec{a}_i, \vec{x} \right\rangle + \sum_{i=1}^{m} \lambda_i c_i = 0$ системы уравнений (1.3) такая, что π$\cap$**V** не имеет целочисленных точек, где $< \vec{a}_i, \vec{x} >$ — скалярное произведение $\vec{a}_i \equiv ($ $a_{i1}\ a_{i2}, ..., a_{in})$ и $\vec{x} \equiv (x_1, x_2, ...x_n)$.

Предложенный алгоритм с помощью λ-теста генерирует множество линейных комбинаций гиперплоскостей. Затем применяется IR-тест для нахождения гиперплоскости из данного множества, которая не имеет целочисленных точек пересечения с **V**.

Алгоритм.

Вход: система уравнений (1.3) и ограничения (1.4), где *n* — количество переменных и *m* — количество уравнений системы.

Выход: **НЕТ ЗАВИСИМОСТИ:** **система** уравнений (1.3) с ограничениями (1.4) не имеет целочисленных решений,
или
ЕСТЬ ЗАВИСИМОСТЬ: система уравнений (1.3) с ограничениями (1.4) имеет целочисленные решения.

Метод:

функция М_Л_ТЕСТ =

1. **для** i =1,…, C_n^{m-1} **цикл**
2. Вычислить (λ_1, λ_2, ..., λ_m);
3. Сгенерировать новую гиперплоскость.
4. Применить IR**-**тест к гиперплоскости.
5. **если** (IR-тест дает ответ **NO**), **то**
6. **возврат НЕТ ЗАВИСИМОСТИ;**
7. **все;**
8. **все;**
9. **возврат ЕСТЬ ЗАВИСИМОСТЬ;**

все.

3.1. Сравнение результатов

Проведем сравнение результатов алгоритма с результатами наиболее известных алгоритмов, таких как НОД-тест, Банержи-тест и λ-тест.

Рассмотрим пример:

```
        for (i=1;i<=100; i++)
        {
            for (j=1;j<=100; j++)
            {
S1:                 A[2*i][ 2*i +3*j]= A[i+j][j-i+6];
            }
        }
```

В этом примере оператор **S1** обращается к элементам массива **A**. Если **S1** не имеет зависимости по данным, то оба цикла в примере можно распараллелить.

Экземпляры оператора **S1:** S1(x_1, x_2) и S1(x_3, x_4) будут обращаться к одной ячейке памяти тогда и только тогда, когда следующая система уравнений имеет целочисленные решения:

$$\begin{cases} 2x_1 - x_3 - x_4 = 0, \\ 2x_1 + 3x_2 + x_3 - x_4 = 6 \end{cases} \qquad (3.1.1)$$

где $1 \le x_1, x_3, x_2, x_4 \le 100$.

Сначала попытаемся разрушить потенциальную зависимость с помощью стандартных тестов. Отметим, что обычно на практике в многомерных массивах каждая размерность тестируется отдельно.

Если применить НОД-тест к первому уравнению (3.1.1), он показывает зависимость, поскольку НОД(2, 0, –1, –1) = 1 и 0 делится на единицу. Во втором уравнении (3.1.1) НОД также равен 1. НОД-тест быстрый тест, но на практике не эффективен, так как в большинстве случаев НОД(a_1, a_2, $...a_n$) = 1.

Аналогично тест Банержи показывает зависимость, потому что оба уравнения имеют вещественное решение в области пространства итераций, т.е. $-198 \le 0 \le 198$ для первого уравнения (4.1) и $-94 \le 6 \le 599$ — для второго.

Как было выше отмечено, λ-тест предназначен для многомерных массивов. В случае, когда анализируются двухмерные массивы и учитываются только границы циклов, множество значений (λ_1, λ_2, ..., λ_m) вычисляется по нижеприведенному определению [7].

Определение 3. Дано уравнение вида $a\lambda_1 + b\lambda_2=0$, где a, b — одновременно не равны нулю, каноническое решение уравнения определяется следующим образом:

$(\lambda_1, \lambda_2) = (1, 0)$, если $a = 0$;
$(\lambda_1, \lambda_2) = (0, 1)$, если $b = 0$;
$(\lambda_1, \lambda_2) = (b, -a)$, если ни один из a, b ненулевой и $b > 0$;
$(\lambda_1, \lambda_2) = (-b, a)$, если ни один из a, b ненулевой и $b < 0$.

С помощью определения 3 вычисляем $\Lambda=\{(2, -2), (3, 0), (1, 1), (1, -1)\}$. Каноническое решение (2, –2) определяет λ-плоскость, которая является линейной комбинацией, и в результате тестирования λ-тест показывает зависимость по данным. Это означает, что гиперплоскость имеет вещественные решения в выпуклом множестве **V**, определяемом границами циклов. Аналогично все λ-плоскости тоже показывают зависимость по данным. Этот случай показывает, что λ-тест не всегда точен.

И наконец, проанализируем данный пример с помощью модернизированного λ-теста. Напомним, что этот алгоритм тоже определяет λ-плоскости, которые являются точной линейной комбинацией, но к ним применяется более точный IR-тест. В нашем примере первое каноническое решение (2, -2) определяет λ-плоскость, т.е. $-6x_2 - 4x_3 = -12$, где $1 \le x_2, x_3 \le 100$. Применим IR-тест к данной λ-плоскости. В результате сложений некоторые коэффициенты исключаются, и поэтому мы сокращаем интервал решений только по осям x_2 и x_3 . Проектируя уравнение на ось x_2, получаем $x_2 = (-4/6)x_3 + (12/6)$, где $1 \le x_2 \le 100$, $1 \le x_3 \le 100$.

Верхняя и нижняя границы соответственно равняются:

$$l_1^{(1)} = (-4/6)^+(1) - (-4/6)^-(100) + (12/6) = (-388/6)$$
$$u_1^{(1)} = (-4/6)^+(100) - (-4/6)^-(1) + (12/6) = (8/6).$$

Эффективным интервалом решений x_2 становится:

$$\left[(-388/6):(8/6)\right] \cap \left[1:100\right] = \left[1:8/6\right].$$

Отсюда, $x_2 \in [\lceil 1 \rceil : \lfloor 8/6 \rfloor]=[1:1]$, т.е. $l_1^{'(1)}=1$, $u_1^{'(1)}=1$.

Так как этот интервал не пуст, спроектируем уравнение $-6x_2 - 4x_3 = -12$ с интервалом $1 \le x_2 \le 1$ на ось x_3. Эффективным интервалом решений по x_3 является [6/4:6/4]. Так как $\lceil 6/4 \rceil > \lfloor 6/4 \rfloor$, то уравнение не имеет целочисленных решений в данном интервале, и оператор **S1** не имеет зависимости по данным.

Рассмотрим еще один пример:

```
          for (i=0; i<=10; i++)
          {
              for (j=0;j<=10; j++)
              {
S1:                   A[j-i+5][i-j+20]= i+j;
S2:               C[i][j]= A[2*i+4*j][7*i+6*j+2];
              }
          }
```

Здесь, если не имеется никакой зависимости по данным между операторами **S1** и **S2**, то оба цикла в примере можно распараллелить. Пусть $x_1 = i$ и $x_2 = j$ для оператора **S1**, и $x_3 = i$ и $x_4 = j$ для **S2**. Уравнения зависимости выглядят следующим образом:

$$\begin{cases} -x_1 + x_2 - 2x_3 - 4x_4 = -5 \\ x_1 - x_2 - 7x_3 - 6x_4 = -18 \end{cases}, \qquad (3.1.2)$$

где $0 \le x_1, x_3, x_2, x_4 \le 10$. Зависимость по данным существует между **S1** и **S2** тогда и только тогда, когда система уравнений (3.1.2) имеет общие целочисленные решения в пределах границ цикла.

Для этого примера стандартные тесты показывают зависимость по данным. Модифицированный тест разрушает потенциальную зависимость. В нашем примере первое каноническое решение (1, 1) определяет λ-плоскость, т.е. $-9x_3 - 10x_4 = -23$, где $0 \le x_3, x_4 \le 10$. Применяем IR-тест к данной λ-плоскости. В результате сложения некоторые коэффициенты исключаются, и поэтому интервал решений сокращается только по осям x_3 и x_4. Проектируя уравнение на ось x_3, получаем $x_3 = (-10/9)x_4 - (23/9)$, где $0 \le x_3 \le 10$, $0 \le x_4 \le 10$.

Верхняя и нижняя границы соответственно равняются:

$$l_1^{(1)} = (-10/9)^+(0)_2 - (-10/9)^-(10) + (23/9) = (-77/9)$$

$$u_1^{(1)} = (-10/9)^+(10) - (-10/9)^-(0) + (23/9) = (23/9).$$

Эффективный интервал решений x_3:

$$[(-77/9):(23/9)] \cap [0:10] = [0:23/9].$$

Отсюда $x_3 \in [\lceil 0 \rceil : \lfloor 23/9 \rfloor] = [0:2]$, т.е. $l_1^{'(1)} = 0$, $u_1^{'(1)} = 2$.

Так как этот интервал не пуст, спроектируем уравнение $-9x_3 - 10x_4 = -23$ с интервалом $0 \le x_3 \le 2$ на ось x_4. Выполняя алгоритм IR-теста шаг за шагом, на некотором шаге получаем интервал решений по x_3: $l_1^{'(1)} = 2$, $u_1^{'(1)} = 1$. Это означает, что уравнение не имеет целочисленных решений в данном интервале, и операторы **S1** и **S2** не имеют зависимости по данным.

Таким образом, алгоритм модифицированного λ-теста может разрушить ложные зависимости, где обычный λ-тест принимает консервативное решение о существовании зависимости по данным.

3.2. Временная сложность

Модифицированный λ-тест включает в себя следующие два этапа: (1) вычисление значений λ и (2) тестирование каждой λ-плоскости. По определению 3, вычисление значений λ имеет временную сложность O(y), где y — константа [4]. Сгенерированная λ-плоскость, тестируется **IR**-тестом. Наихудшая временная сложность **IR**-теста: $O(kz)$, где $k = \min\{u_i - l_i + 1 : 1 \le I \le z\}$, z — число переменных в сгенерированной λ-плоскости [8]. Следовательно, модифицированный λ-тест имеет, в общем случае, наихудшую временную сложностью O(C_n^{m-1} * (k$z + y$)).

В случае анализа двухмерного массива, по Теореме 2 из [7], временная сложность модифицированного λ-теста: O(n * ($kz + y$)).

ЗАКЛЮЧЕНИЕ

При распараллеливании программы одной из основных проблем является выявление зависимости по данным. Особенно вызывает трудности анализ многомерных массивов. В обычной практике каждая размерность массивов тестируется индивидуально. Но имеются алгоритмы, которые специально предназначены для многомерных массивов, например λ-тест.

В данной работе представлен алгоритм модифицированного λ-теста, в котором λ-тест интегрирован с точным IR-тестом. Экспериментальные сравнения результатов показали, что модифицированный алгоритм более точен по сравнению с НОД, Банержи и λ-тестами на зависимость. Таким образом, модифицированный λ-тест является точным тестом для выявления зависимости по данным в многомерных массивах, содержащих сцепленные индексы.

В данное время алгоритм применяется только в тех ситуациях, где границы циклов постоянные и индексные выражения линейны. Дальнейшей целью является расширение теста с учетом более общих критериев, в которых границы циклов являются функциями индексов внешних циклов, а также тестирование зависимости по всем измерениям векторов направлений.

Практическим результатом данной работы является тест, который может быть использован в блоке анализа зависимостей по данным в проектируемой системе быстрого прототипирования компилятора.

СПИСОК ЛИТЕРАТУРЫ

1. Евстигнеев В.А. Анализ зависимостей: состояние проблемы // Системная информатика: Сб. науч. тр. — Новосибирск: Наука, 2000. — Вып. 7. — С. 112–173.
2. Banerjee U. Loop transformations for restructuring compilers: the Foundations. — Boston: Kluwer Academic Publishers, 1993.
3. Kong X., Klappholz D., and Pssaris K. The I Test: An Improved Dependence Test for Automatic Parallelization and Vectorization // IEEE Transactions on Parallel and Distributed Systems. — 1991. — Vol. 2(3) . — P. 342–349.

4. Chang, W.-L., Chu, C.-P., Wu, J. A multi-dimensional version of the I test // Parallel Computing. — 2001. — Vol. 27. — P. 1783–1799.
5. Wolfe M., and Tseng C. The Power Test for Data Dependence // IEEE Transactions on Parallel and Distributed Systems. September 1992.
6. Pugh W. The Omega test: a fast and practical integer programming algorithm for dependence analysis // Communications of the ACM. — 1992. — Vol. 35(8) . — P.102–114.
7. Li Z., Yew P.-C., Zhu C.-Q. An efficient data dependence analysis for parallelzing compilers // IEEE Transaction on Parallel and Distributed Systems. — 1990. — Vol. 1(1) . — P. 26–34.
8. Huang T.-C., Yang C.-M. Data dependence analysis for array references // J. of Systems and Software. — 2000. — Vol. 52. — P. 55–65.
9. Shen Z., Li Z., Yew P.-C. An empirical study of Fortran programs for parallelizing compilers // IEEE Transaction on Parallel and Distributed Systems. — 1992. — Vol. 1 (3) . — P. 356–364