Российская академия наук
Сибирское отделение
Институт систем информатики
имени А. П. Ершова

Р.Н. Арапбаев, В.А. Евстигнеев, Р.А. Осмонов

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ТЕСТОВ НА ЗАВИСИМОСТЬ ПО ДАННЫМ

Препринт
141

Новосибирск 2006

В работе представлен сравнительный обзор тестов на зависимость по данным, применяемых в распараллеливающих компиляторах. Даны сопоставления сильных и слабых сторон тестов как на примерах, так и по оцениваемым характеристикам отдельных критериев.

Ключевые слова: распараллеливающие компиляторы, зависимость по данным, распараллеливание циклов, оптимизация, линейное диофантово уравнение.

Данная работа частично поддерживалась грантом РФФИ № 05-01-00816.

Siberian Division of the Russian Academy of Sciences
A. P. Ershov Institute of Informatics Systems

Vladimir A. Evstigneev, Ruslanbek N. Arapbaev, Rafhat A. Osmonov

COMPARATIVE ANALYSIS OF TESTS FOR DATA DEPENDENCE

Preprint
141

Novosibirsk 2006

In the paper, we present a comparative review of tests for data dependence that are applied in parallelizing compilers. Comparisons of power and limits of tests characteristics of different criteria are given both on examples and under estimated.

Key words: parallelizing compilers, data dependence, loop parallelization, optimization, linear Diophantine equation.

1. ВВЕДЕНИЕ

Данная работа является продолжением работы [1], описывающей современное состояние проблемы анализа зависимостей при автоматическом распараллеливании последовательных программ. Автоматическое распараллеливание является одним из основных средств разработки параллельных программ. Извлечение скрытого параллелизма, в первую очередь, связано с анализом циклов и заключается в нахождении зависимостей между итерациями цикла. Мощность распараллеливающих компиляторов зависит от блока анализа зависимостей, который состоит из нескольких серий тестов на зависимость. Рассмотрим гнездо циклов с индексными переменными $i_1, i_2, \ldots, i_k$.

```
do  i1= L1, U1
          do  i2= L2, U2
                …
          do  ik= Lk, Uk
< S1>     X [h1(i1, i2, …, ik), h2(i1, i2, …, ik), ..., hd(i1, i2, …, ik)] = ...
< S2>     ... = X [g1(i1, i2, …, ik), g2(i1, i2, …, ik), ..., gd(i1, i2, …, ik)]
          enddo
                …
          enddo
enddo
```

Традиционно под зависимостью по данным понимается зависимость, связанная с совпадением ссылок на элементы массивов. Пусть операторы S_1 и S_2 обращаются к массиву ***X,*** индексные выражения которого представлены линейными функциями $h_q(i_1, i_2, \ldots, i_k)$ и $g_q(i_1, i_2, \ldots, i_k)$, где $q = 1,\ldots,d$. Зависимость по данным между операторами S_1 и S_2 имеется тогда и только тогда, когда здесь существуют целые $i_1, i_2, \ldots, i_k$ и $j_1, j_2, \ldots, j_k$, такие что:

$$\begin{aligned} h_1(i_1, i_2, \ldots, i_k) &= g_1(j_1, j_2, \ldots, j_k) \\ h_2(i_1, i_2, \ldots, i_k) &= g_2(j_1, j_2, \ldots, j_k) \\ &\ldots \\ h_d(i_1, i_2, \ldots, i_k) &= g_d(j_1, j_2, \ldots, j_k) \end{aligned} \qquad (1)$$

и

$$i_p, j_p \in [L_p, U_p], \text{ где } p=1,\ldots,k \qquad (2)$$

(L_p, U_p могут зависеть от индексов верхних циклов $i_1, i_2, \ldots, i_{p-1}$).

Как отмечено в [1], прямой подход к решению задачи выявления зависимостей в общем случае невозможен, так как даже в линейном случае это приводит к NP-полной проблеме отыскания целочисленного решения системы диофантовых уравнений. Для преодоления этой проблемы к настоящему времени разработаны многие тесты, дающие приближенные решения задачи. Среди них на практике наибольшее распространение получили НОД-тест и тест на основе неравенства Банержи [2–4]. Среди других укажем Интервальный тест (I-тест) [5], λ-тест [6], дельта-тест [7], Power-тест [8], Омега-тест [9], Ипсилон-тест [10], IR-тест ("interval reduction") [11] и др.

Эти тесты используют различные математические инструменты, и каждый из них имеет свои преимущества и недостатки. Значит, некоторые тесты более эффективны, но менее точны, а другие более точны, но не эффективны при применении. Поэтому на практике используется алгоритм зависимости по данным, который состоит из серии тестов в определенном иерархическом порядке использования. Например, в проекте SUIF Стенфордского университета алгоритм состоит из серии точных тестов [12], где последним тестом служит метод исключения Фурье–Моцкина. В системе Parafrase-2 [13] используется стратегия применения НОД-теста и теста Банержи, а в распараллеливающем компиляторе INRIA PIAF [14] внедрена стратегия на основе применений методов целочисленного и линейного программирования и др. Но вопрос, какая последовательность или стратегия лучшая, до сих пор остается открытым.

Цель данной работы — изучить основные тесты на зависимость по данным, дать сопоставление их сильных и слабых сторон. Эти результаты могли бы использоваться для выработки стратегий применения тестов при выявлении зависимости по данным в блоке анализа зависимостей.

По применению различных математических инструментов, тесты на зависимость можно условно разделить на 3 группы.

1. *Приближенные тесты.* Например, НОД-тест, тест Банержи и их расширений, Обобщенный НОД-тест [4], Дельта-тест, I-тест, λ-тест для многомерных массивов и др.
2. *Точные тесты.* Тесты, использующие алгоритм удаления Фурье–Моцкина [15]: Power-тест, Омега-тест, MHL-тест ("Maydan D., Hennessy J., and M. Lam") [12] и др.
3. *Точные тесты, основанные на методах линейного и целочисленного программирования,* например СМ-тест [16], использующий симплекс метод и др.

Отметим, что кроме этих тестов имеются тесты, которые используют *теоретико-графовый* метод для решения систем линейных неравенств. Например, алгоритм Шостака [17], LR-тест (“Loop Residue”) [12], и др.

Как сказано выше, тесты на зависимость по данным имеют два показателя. Первый показатель, которому всегда придавалось особое значение — это *скорость* теста. Второй показатель — это требуемая *степень точности* теста. Точность определяется по трем пунктам.

1. Точность зависит от ответа, который мы хотим получить, например, некоторые тесты определяют векторы направлений цикла и для этого дополнительные ограничения должны быть добавлены к исходной системе.
2. Точность зависит от ответа, выдаваемого выполненным алгоритмом. В самом деле, три ответа являются допустимыми:
   - **No,** нет решения. Система является несовместимой, и это доказывает, что здесь нет никаких зависимостей;
   - **Yes,** имеется решение. Операторы являются зависимыми;
   - **Maybe,** тест не может доказать или опровергнуть существование решений.

   В первых двух случаях выводится правильное решение. Результат является *точным*. А в последнем случае он *неточный*. В этом случае тесты тоже должны показывать существование зависимости, чтобы не допускать неправильных преобразований программ.
3. Точность зависит от знания, которое мы имеем о семантике данных. Существуют достаточно сложные методы в области потокового анализа программ, которые влияют на качество анализа зависимостей по данным. Например, протягивание констант, символический анализ и др.

Работа организована следующим образом. В разд. 2 описываются основные определения анализа зависимостей по данным. В разд. 3 приводится классификация тестов на зависимость, рассматриваются основные механизмы и характеристики. В разд. 4 приводится заключение и список литературы по данной тематике.

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

2. МОДЕЛЬ ПРОГРАММЫ

Будем рассматривать зависимости, порождаемые ссылками на элементы массива, имеющиеся в теле DO-цикла, поскольку при анализе зависимостей по данным решение этой задачи представляет основную трудность.

В качестве модели программы рассматривается нормализованное совершенное гнездо циклов, (тело циклов находится внутри самого внутреннего цикла) такое, что приращения индексных переменных равны 1.

Наконец, предполагается, что индексные выражения — линейные с целочисленными коэффициентами, а границы DO-циклов — известные константы или также могут линейно зависеть от индексной переменной любого внешнего цикла.

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) может быть записана в следующем виде:

$$\begin{aligned} &a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n = a_{1,0} \\ &a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_n = a_{2,0} \\ &\qquad\qquad \ldots\ldots\ldots\ldots \\ &a_{m,1}x_1 + a_{m,2}x_2 + \ldots + a_{m,n}x_n = a_{m,0} \end{aligned} \qquad (3)$$

и при условии выполнения следующих систем ограничений, включая границы циклов и другие факты, которые могут быть собраны из текста программы:

$$L_i \le x_i \le U_i, \quad \text{где } i=1,\ldots,n. \qquad (4)$$

Будем оценивать характеристики тестов на зависимость по следующим критериям:

1) *допустимая форма коэффициентов индексных переменных,*
2) *допустимая форма границ циклов,*
3) *степень использования тестом системы ограничений* (4)*,*
4) *условия, при которых тест является точным (если это вообще возможно),*
5) *вычислимость тестом дистанционных векторов зависимости,*
6) *возможность решения тестом системы уравнений* (3) *одновременно (или каждого уравнения в отдельности),*

7) *сложность алгоритма,*
8) *уместность использования теста во время выполнения компиляции.*

3. ТЕСТЫ НА ЗАВИСИМОСТЬ ПО ДАННЫМ

3.1. Основные методы

3.1.1. НОД-тест

НОД-тест (“GSD-test”) разработан одним из первых [2]. Данный тест является одним из важных тестов на зависимость по данным, и часто включается в состав других алгоритмов анализа зависимости как начальный этап для выявлений зависимостей.

Он обрабатывает каждое уравнение из системы уравнений (3) отдельно. Тест основывается на теореме из элементарной теории чисел, которая утверждает, что отдельное уравнение

$$a_1 x_1 + a_2 x_2 + \ldots + a_n x_n = a_0 \qquad (5)$$

имеет целочисленное решение тогда и только тогда, когда наибольший общий делитель коэффициентов уравнения $a_1, a_2, \ldots, a_n$ есть делитель a_0. Тест заключается в проверке указанной делимости.

Заметим, что НОД-тест не проверяет существование решения, которое одновременно целочисленно и удовлетворяет ограничениям на область изменения переменных. НОД-тест совершенно игнорирует границы изменения переменных и определяет, имеет ли уравнение решение для любых целочисленных значений всех переменных. Таким образом, НОД-тест порождает ложные зависимости, когда имеется решение за пределами границ, но нет целочисленного решения внутри границ. Чтобы показать возможности теста, рассмотрим пример.

Пример 1.

```
        do  i= 1, N
S1:             A(2i) =B(i) + C(i)
S2:             D(i) = A(2i+1)
        enddo
```

Для двух ссылок на массив $A(2i)$ и $A(2i+1)$ уравнение зависимости имеет вид:

$2x_1 - 2x_2 = 1$

и НОД (2,2) = 2,

так как 2 не делит 1, то зависимости нет.

Таким образом, НОД-тест не может доказать существование зависимости, но он полезен при его опровержении. Следовательно, нет такой ситуации, в которой НОД-тест является точным.

Форма коэффициентов: целочисленная константа
Форма границ циклов: никогда не используется
Использование ограничений: нет
Точность: неточный
Дистанционные вектора: нет
Одновременное решение: нет
Сложность: линейная
Уместность использования: не дорогой. Недостаток точности — серьезное ограничение.

3.1.2. Тест Банержи [2]

В отличие от НОД-теста тест Банержи принимает во внимание границы циклов, но устанавливает наличие не целочисленного, а вещественного решения уравнения зависимости. Тест дает неверные результаты, когда уравнение имеет вещественное решение, но не имеет в заданных границах целочисленного решения. Как и в случае НОД-теста это приводит к появлению лишних ограничений потенциального параллелизма. Так как тест Банержи не имеет явных преимуществ перед НОД-тестом, то обычно реализуются сразу оба теста.

Пусть, как выше, существует система уравнений, каждое из которых имеет форму (5):

$$a_1 x_1 + a_2 x_2 + \ldots + a_n x_n = a_0$$

и ограничение на область значений переменных вида :

$$L_i \le x_i \le U_i$$

Пусть a — целое число. Определим

$a^+ = a$, если $a \ge 0$, иначе 0;

$a^- = -a$, если $a \le 0$, иначе 0.

Когда тест Банержи применим, то он вычисляет экстремальные значения

$$\text{low} = \sum_{i=1}^{n} (a_i^+ L_i - a_i^- U_i)$$

и

$$\text{high} = \sum_{i=1}^{n} (a_i^+ U_i - a_i^- L_i).$$

Для левой части уравнения зависимости имеем low ≤ high. По теореме о промежуточном значении уравнение зависимости имеет вещественное решение внутри границ цикла тогда и только тогда, когда

$$\text{low} \le a_0 \le \text{high}.$$

Тест Банержи состоит в проверке этого условия.

Рассмотрим пример.

Пример 2.

```
      do  i= 1, 10
S1:           A(i) =A(7i + 4) – C(i)
      enddo
```

В этом примере оператор **S1** обращается к элементу массива **A**. Если **S1** не имеет самозависимости по данным, то цикл можно распараллелить. Уравнение зависимости имеет следующий вид:

$$x_1 - 7x_2 = 4, \text{ где } 1 \le x_1, x_2 \le 10.$$

НОД-тест в этом случае просто бесполезен, так как наибольший делитель коэффициентов равен 1 и он не может разрушить потенциальной зависимости в цикле.

Применяя тест Банержи, получаем неравенство:

$$-69 \le 4 \le 3.$$

Данное неравенство некорректно, следовательно, в данном примере зависимости по данным отсутствуют.

Тест Банержи принимает во внимание границы циклов только тогда, когда можно установить статически, что они константны. Если границы не могут быть статически определены как константы, то тест Банержи просто не применим.

Однако его эффективность и полноценность при опровержении зависимостей, делают его одним из самых общих тестов, используемых в распараллеливающих компиляторах. В исследованиях [6, 18, 19, 20] показано, что если коэффициенты линейного уравнения удовлетворяют некоторым условиям, например когда коэффициенты в уравнении зависимости равны ±1, то тест Банержи становится точным тестом.

Известны еще два вида теста Банержи: *обобщенный* (или *трапецеидальный* тест) и *неограниченный*. Соответственно первый используется, когда границы некоторых циклов являются функциями индексов внешних циклов, а последний, когда неизвестна верхняя или нижняя границы.

Форма коэффициентов: целочисленная константа
Форма границ циклов: обычно целочисленная константа, но возможно использование символьных выражений
Использование ограничений: да — рассматривает границы циклов
Точность: в ряде случаев точный [6, 18, 19, 20]
Дистанционные вектора: нет
Одновременное решение: нет
Сложность: линейная
Уместность использования: линейная сложность, хотя недостаток точности — серьезное ограничение

3.1.3. Метод исключения переменных Фурье—Моцкина

Данный метод является самой общей техникой для решения систем линейных ограничений. Метод позволяет решать системы линейных уравнений (3) и неравенств (4). Метод исключения переменных Фурье—Моцкина (“FMVE — Fourier—Motzkin variable elimination”) [15] удаляет переменную в задаче линейного программирования. На интуитивном уровне это значит, что данный метод сводит задачу от n-мерной к n–1-мерной.

Рассмотрим два ограничения на x: нижнюю границу $L \le bx$ и верхнюю границу $ax \le U$ (где a и b — положительные целочисленные константы). Мы можем комбинировать эти ограничения, получая $aL \le abx \le bU$. Тенью (shadow) этой пары ограничений служит $aL \le bU$. Метод Фурье—Моцкина вычисляет тень множества ограничений путем комбинирования всех ограничений, которые не включают удаляемую переменную, с результатами каждой комбинации нижней и верхней границ удаляемой переменной. Вещественная тень есть консервативная аппроксимация целочисленной тени множества ограничений.

Таким образом, метод Фурье—Моцкина решает системы линейных равенств и неравенств, исключая переменные по одной. Этот процесс продолжается до тех пор, пока все переменные не будут исключены или пока не будет найдено противоречие. В последнем случае, FMVE объявляет, что зависимости не существует. Иначе, принимается решение, что система имеет действительные решения. В худшем случае, в процессе исключения переменных число неравенств в системе может расти экспоненциально. Однако на практике, многие из произведенных неравенств бывают избыточными, и показательный рост обычно не происходит.

Рассмотрим пример.

Пример 3.

Дано следующее линейное уравнение и множество границ цикла:

$$x + 4y + 7z - 34 = 0, \text{ где : } \quad \begin{array}{l} 0 \le x \le 15 \\ -5 \le y \le 10 \\ 3 \le z \le 12. \end{array}$$

В начале записывается эквивалентная система линейных неравенств:

$$\begin{array}{rcl} 34 \le & x + 4y + 7z & \le 34 \\ 0 \le & x & \le 15 \\ -5 \le & y & \le 10 \\ 3 \le & z & \le 12. \end{array}$$

Так как $a_x = 1$, первым исключается из системы x. Выражаем первое неравенство относительно x:

$$\begin{array}{l} 34 - 4y - 7z \le x \le 34 - 4y - 7z \\ 0 \le x \le 15 \\ -5 \le y \le \quad 10 \\ 3 \le z \le 12. \end{array}$$

Сравнение верхних и нижних границ x дает:

Верхняя граница	*Нижняя граница*	*Комбинация*	*Результат*
$34 - 4y - 7z \le x$	$x \le 34 - 4y - 7z$	$34 - 4y - 7z \le x \le 34 - 4y - 7z$	$0 \le 0$
$34 - 4y - 7z \le x$	$x \le 15$	$34 — 4y - 7z \le x \le 15$	$19 \le 4y + 7z$
$0 \le x$	$x \le 34 - 4y - 7z$	$0 \le x \le 34 - 4y - 7z$	$4y + 7z \le 34$
$0 \le x$	$x \le 15$	$0 \le x \le 15$	$0 \le 15$

Первый и последний результаты непротиворечивые и включают в себя только постоянные элементы, следовательно, они игнорируются. Объединяя оставшихся два результата, получаем неравенство $19 \le 4y+7z \le 34$. При этом добавляем его в систему и удаляем из системы два неравенства, включающие в себя x.

В результате формируется новая система:

$$19 \le 4y + 7z \le 34$$
$$-5 \le y \le 10$$
$$3 \le z \le 12.$$

Исключая y получим следующую систему:

$$-7z + 19 \le 4y \le -7z + 34$$
$$-5 \le y \le 10$$
$$3 \le z \le 12.$$

Сравнение нижних и верхних границ для y:

Верхняя граница	*Нижняя граница*	*Комбинация*	*Результат*
$-7z + 19 \le 4y$	$4y \le -7z + 34$	$-7z + 19 \le 4y \le -7z + 34$	$19 \le 34$
$-7z + 19 \le 4y$	$y \le 10$	$-7z + 19 \le 4y < 40$	$-21 \le 7z$
$-5 \le y$	$4y \le -7z + 34$	$-20 \le 4y \le -7z + 34$	$7z \le 54$
$-5 \le y$	$y \le 10$	$-5 \le y < 10$	$0 \le 15$

Аналогично предыдущему добавляется новое неравенство $-21 \le 7z \le 54$. После нормализации:

$$-3 \le z \le 7$$
$$3 \le z \le 12.$$

Затем исключаем z:

Верхняя граница	*Нижняя граница*	*Комбинация*	*Результат*
$-3 \le z$	$z \le 7$	$-3 \le z \le 7$	$-3 \le 7$
$-3 \le z$	$z \le 12$	$-3 \le z \le 12$	$-15 \le 0$
$3 \le z$	$z \le 7$	$3 \le 7$	$0 \le 4$
$3 \le z$	$z \le 12$	$3 \le z \le 12$	$3 \le 12$

Заключительное неравенство $-15 \le 0 \le 4$. Так как после исключения всех переменных не достигнуто никаких противоречий, делается вывод, что исходная система линейных неравенств имеет действительное решение.

Некоторые точные тесты используют метод Фурье—Моцкина чтобы доказать или опровергнуть существование зависимости. Например, Power-тест [8], Омега-тест [9] и тест Майдана (MHL-тест) [12] и др.

Форма коэффициентов: линейные выражения
Форма границ цикла: линейные выражения
Использование ограничений: да
Точность: точный для вещественного решения
Дистанционные вектора: нет
Одновременное решение: да
Сложность: экспоненциальная, но обычно полиномиальная
Уместность использования: дорогой.

3.2. Расширенные методы

3.2.1. Приближенные тесты

3.2.1.1. Обобщенный НОД-тест. Обобщенный НОД-тест [3,4] решает систему линейных уравнений (3), которую можно записать в матричной форме $\boldsymbol{hA = c}$, где $\boldsymbol{A}$ является матрицей, сформированной из коэффициентов системы, $\boldsymbol{h}$ — вектор переменных в системе, и $\boldsymbol{c}$ — вектор свободных членов. Тест определяет, существуют ли одновременные целочисленные решения системы. Сначала тест, используя свойство линейной алгебры, формирует ($n \times m$) матрицу коэффициентов $\boldsymbol{A}$ и ($1 \times m$) матрицу коэффициентов $\boldsymbol{c}$, где n — количество переменных в системе, m — количество уравнений. Затем определяется *унимодулярная*[*] матрица преобразования $\boldsymbol{U}$, чтобы преобразовать $\boldsymbol{A}$ в верхнюю треугольную («*эшелонную*») матрицу $\boldsymbol{D}$ с помощью элементарных строковых операций преобразования (как в исключении Гаусса). Если существует целочисленное решение $\boldsymbol{t}$ такое, что $\boldsymbol{tD = c}$, то целочисленные решения существуют и зависимость допускается. Так как $\boldsymbol{D}$ — верхняя треугольная матрица, можно использовать простую обратную замену, чтобы легко определить, существует ли решение $\boldsymbol{t}$. Произведение матриц $\boldsymbol{tU}$ позволяет вычислить дополнительную информацию в виде постоянных дистанционных векторов. Хотя обобщенный НОД-тест дает ответ на существование целочисленного решения, но в нем не учитываются ограничения на область изменения переменных. Поэтому обобщенный НОД-тест не может доказать существование зависимости, но полезен при ее опровержении.

[*] Унимодулярной матрицей является матрица, модуль детерминанта которой равен 1.

Рассмотрим пример:

Пример 4.

```
  do i = 1, 100
       do j = 25, i + 50
S1:          B(i + j – 50, i + 4j – 100)=...
S2:          ... = ... B(200 – i – 2j, 500 – i – 5j) ...
       enddo
  enddo
```

Гнездо цикла имеет следующую систему уравнений зависимости

$$x_1 + x_2 + x_3 + 2x_4 = 250$$
$$x_1 + 4x_2 + x_3 + 5x_4 = 600.$$

Матричная форма уравнений зависимости $\boldsymbol{hA = c}$ имеет вид:

$$\left(x_1, x_2, x_3, x_4\right) \begin{pmatrix} 1 & 1 \\ 1 & 4 \\ 1 & 1 \\ 2 & 5 \end{pmatrix} = (250, 600).$$

Дополним $\boldsymbol{A}$ единичной матрицей, чтобы получить $\boldsymbol{UD}$:

$$\left(\begin{array}{cccc|cc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 4 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 2 & 5 \end{array}\right).$$

($\boldsymbol{U}$ включает в себя первые четыре столбца, $\boldsymbol{D}$ — конечные два.) Чтобы преобразовать $\boldsymbol{D}$ в *эшелонную* матрицу, выполняются элементарные строковые операции преобразований, в результате получаем:

$$\left(\begin{array}{cccc|cc} 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & -2 & 1 & 0 & 3 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & -1 & 0 & 0 \end{array}\right).$$

Можно проверить, что ***UA = D***. Затем решается уравнение ***tD = c***,

$$\left(t_1,\ t_2,\ t_3,\ t_4\right)\begin{pmatrix}1 & 1\\ 0 & 3\\ 0 & 0\\ 0 & 0\end{pmatrix} = (250, 600)\ .$$

Общее решение уравнения $\left(t_1,\ t_2,\ t_3,\ t_4\right)\boldsymbol{D} = (250, 600)$ — (250, 350/3, t_3, t_4), где t_3 и t_4 являются неопределенными. Так как нет никакого целочисленного решения, обобщенный НОД-тест показывает, что нет никакой зависимости между операторами **S1** и **S2**.

Обобщенный НОД-тест часто объединяется с другими тестами, которые применяют различные тесты к ***t*** вектору. Типичный тест, принадлежащий к этому классу тестов — Power-тест [8]. Другие тесты этого типа — ***SVPC***-тест, ***Acyclic*** -тест, и ***LR-***тест [12].

Форма коэффициентов: целочисленная константа
Форма границ циклов: линейные выражения
Использование ограничений: да
Точность: точный до степени, сколько система ограничений позволяет
Дистанционные вектора: иногда
Одновременное решение: да
Сложность: показательная, в худшем случае — полиномиальная
Уместность использования: дорогой.

3.2.1.2. λ-тест. λ-тест предназначен для *сцепленных* и *многомерных* ссылок массивов. Он исследует систему равенств (3) и неравенств (4) и определяет, имеет ли система действительные решения [6].

Геометрически каждое линейное уравнение в (3) представляет собой гиперплоскость π в пространстве $\mathbf{R^n}$. Пересечения **m** гиперплоскостей ***S*** соответствуют общим решениям системы (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 (3) пересекает **V**, существует вероятность, что **S** не пересечет **V**. Если можно найти новую гиперплоскость, которая содержит **S**, но не пересекает **V**, то это доказывает, что **S** и **V** не пересекаются. 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 (3):

$\left\langle \sum_{i=1}^{m} \lambda_i \vec{a}_i, \vec{x} \right\rangle + \sum_{i=1}^{m} \lambda_i c_i = 0$, где $< \vec{a}_i, \vec{x} >$ — скалярное произведение векторов

$\vec{a}_i \equiv (a_{i1}\, a_{i2}, \ldots, a_{in})$ и $\vec{x} \equiv (x_1, x_2, \ldots x_n)$ [6].

С другой стороны, если **S** и **V** пересекутся, то никакая такая линейная комбинация не существует.

Массив (λ_1, λ_2, ..., λ_m) 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 [6], в общем случае λ-тест генерирует C_n^{m-1} таких гиперплоскостей, которые являются линейной комбинацией (3) и называются λ-плоскостями. Чтобы определить пересекает ли каждая λ-плоскость **V**, применяется тест Banerjee—Wolfe 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.

Рассмотрим пример.

Пример 5.

```
do i = 1, 100
    do j = 1, 100
S1:             A(2i+3j+4)( 3i + j + 7) = A(i – j + 5)( 2i – j + 6);
    enddo
enddo
```

Система уравнений зависимости имеет следующий вид:

$$2x_1 + 3x_2 - x_3 + x_4 = 1 \quad ,$$
$$3x_1 + x_2 - 2x_3 + x_4 = -1 \ ,$$

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

Сначала попытаемся разрушить потенциальную зависимость с помощью стандартных тестов. Отметим, что обычно на практике многомерных массивов тестируют каждую размерность отдельно.

Если к первому уравнению системы применяется НОД-тест, то он показывает зависимость, поскольку НОД(2, 0, –1, –1) = 1 и 0 делится на единицу. Во втором уравнении тоже НОД равен единице.

Аналогично тест Банержи показал зависимость, потому что оба уравнения имеют вещественные решения в области итерационного пространства, т.е. $-94 \le 1 \le 599$ для первого и $-195 \le -1 \le 498$ для второго уравнения.

Как выше сказано, λ-тест предназначен для многомерных массивов. В случае, когда анализируются двухмерные массивы и учитываются только границы циклов, имеем не более **n** гиперплоскостей, которые соответствуют линейной комбинации системы уравнений зависимости**.**

Используя определение из [6], вычисляется первое каноническое решение (3, –2), которое определяет λ-плоскость, т.е. $7x_2 + x_3 + x_4 = 5$, где $1 \le x_2, x_3, x_4 \le 100$. Затем применяется тест Банержи к этой λ-плоскости. В результате неравенство Банержи не выполняется ($9 \le 5 \le 900$), и λ-тест сообщает, что оператор **S1** не имеет зависимости по данным от себя.

Так как λ-тест является многомерным вариантом теста Банержи, в работах [21, 22] представлены так называемые *Обобщенный λ-тест* и *неограниченный λ-тест*, соответственно в них использовался *Обобщенный* и *неограниченный* варианты теста Банержи.

Форма коэффициентов: линейные выражения
Форма границ циклов: целочисленные константы
Использование ограничений: да, рассматривает границы цикла подобно тесту Банержи
Точность: не точный для больше чем двухмерных индексных выражений
Дистанционные вектора: да
Одновременное решение: да
Сложность: полиномиальная
Уместность использования: обычно дорогой.

3.2.1.3. I-тест (Интервальный тест) [5]. Данный тест является комбинацией тестов НОД и Банержи. Как и НОД-тест, он проверяет существование целочисленного решения; как и тест Банержи, он учитывает ограничения индексной переменной. При этом I-тест можно применять при таких огра-

ничениях, которые недостаточны для применения теста Банержи. I-тест преобразует уравнение зависимости (5) в интервальное уравнение:

$$a_1 x_1 + a_2 x_2 + \ldots + a_n x_n = [L, U]. \qquad (6)$$

В правой части уравнения (6) L, U верхняя и нижняя границы, являющиеся константами. В исходной форме верхняя и нижняя границы равны постоянному значению в правой части уравнения (5). Тест, в каждой итерации выбирая переменную из левой части уравнения, перемещает ее в верхнюю и нижнюю границы в правой части (используя механизм *теста Банержи*), затем применяет НОД-тест к оставшимся коэффициентам. Этот процесс продолжается, пока или он может быть доказан, что уравнение является недопустимым, или нет больше переменных, которые могут быть перемещены. Подробное теоретическое объяснение I-теста и некоторые преобразование интервальных уравнений описано в [1].

Рассмотрим пример.

Пример 6.

Интервальное уравнение, преобразованное из уравнений зависимости:

$$2x_1 - 6x_2 + 14x_3 = [16, 16]$$

с ограничениями на переменные: $1 \le x_1 \le 3$, $1 \le x_2 \le 2$, $1 \le x_3 \le 4$. Тесты НОД и Банержи показывают в этом случае существование зависимости. Сначала нормализуем уравнение:

$$x_1 - 3x_2 + 7x_3 = [8,8].$$

НОД(1, –3, 7) = 1 делит 8, поэтому НОД-тест показывает существование зависимости. Экстремальные значения выражения $x_1 - 3x_2 + 7x_3$ при данных ограничениях на переменные x_1, x_2 и x_3, вычисленные тестом Банержи: 2 и 28; так как $2 \le 8 \le 28$, то тест Банержи тоже покажет существование решения уравнения зависимости. Перемещая переменную x_1 в правую часть уравнения, получим:

$$-3x_2 + 7x_3 = [8 - 3, 8 - 1] = [5, 7].$$

Интервальный НОД-тест, по-прежнему, показывает существование зависимости.

Но теперь длина интервала справа увеличена втрое, переносим вправо выражение $-3x_2$.

$$7x_3 = [5 + 3, 7 + 6] = [8, 13].$$

Применение интервального НОД-теста на этой стадии показывает отсутствие зависимости.

В [23] I-тест расширен для анализа зависимостей по данным с учетом вектора направления зависимости (*DVI-тест*). В работах [24, 25] представлен многомерный вариант I-теста, где I-тест интегрирован с многомерным λ-тестом.

Форма коэффициентов: целочисленная константа
Форма границ циклов: целочисленная константа
Использование ограничений: да, границы циклов
Точность: точный для случаев, где все коэффициенты равны ± 1 [18, 19, 20]
Дистанционные вектора: нет
Одновременное решение: нет
Сложность: линейная
Уместность использования: недорогой.

3.2.1.4. Дельта-тест. Главным недостатком точных тестов является их стоимость выполнения, т.е. скорость теста. Чтобы облегчить эту проблему, был разработан Дельта-тест [7] для определенных классов ссылок массива, которые часто встречаются в научных программных кодах. Дельта-тест сначала классифицирует индексные выражения массивов на следующие категории: ZIV (нулевая индексная переменная), SIV (единственная индексная переменная) и MIV (составная индексная переменная) формы. Соответственно к каждой форме применяется одноименные тесты. ZIV и SIV формы подразделяется соответственно в различные формы, которые легко и точно решаются.

ZIV-тест — это тест на зависимость, который принимает на вход два циклически инвариантных (loop-invariant) выражения. Если тест определяет, что эти два выражения не могут быть равны, то зависимости не существует. Тест ZIV может быть легко расширен для символических выражений. Просто формируется выражение, представляющее разницу между двумя индексными выражениями; если разница упрощается до ненулевой константы, это доказывает независимость.

SIV-тест применяется, когда в индексном выражении массива используется одна индексная переменная. SIV-формы индексных выражений делятся на две категорий: *сильные* и *слабые*.

Говорят что индексное выражение SIV для индекса *i*, называется *сильным*, если оно имеет форму ($ai + c_1$, $ai' + c_2$), т.е. если оно линейное, коэффициенты двух экземпляров индекса *i* являются константами и равны. Для

сильных индексных выражений SIV, *расстояние зависимости* определяется следующем образом:

$$d = i' - i = \frac{c_1 - c_2}{a}.$$

Зависимость существует тогда и только тогда, когда d — целое число и $|d| \le U - L$, где U и L — верхние и нижние границы цикла. Для зависимостей, которые действительно существуют, *направление зависимости* определяется как:

$$направление = \begin{cases} < & if \quad d > 0 \\ = & if \quad d = 0 \\ > & if \quad d < 0 \end{cases}.$$

Таким образом, сильный SIV-тест является точным и эффективным тестом.

Рассмотрим пример.

Пример 7.

```
        do i=1,10
S1:       A(3i+4)=A(3i-2)
        enddo
```

Расстояние зависимости: $d = 2$ и $|d| \le 10 - 1$, следовательно, получаем точный ответ о существование зависимости по данным.

Слабое индексное выражение SIV имеет форму $(a_1 i + c_1,\ a_2 i' + c_2)$, где $a_1 \ne a_2$. Рассмотрим два частных случая. Случай, где $a_1 = 0$ или $a_2 = 0$, называется *слабо-нулевым* индексным выражением *SIV* («weak-zero SIV»). Если $a_2 = 0$, то уравнение зависимости принимает вид:

$$i = \frac{c_2 - c_1}{a_1}.$$

Просто необходимо проверить, что результирующее значение для i является целым числом и находится в пределах границ цикла. Подобная проверка применяется, когда $a_1 = 0$.

Когда $a_2 = -a_1$, все индексные выражения маркируются как *слабо-пересекающиеся SIV* (“weak-crossing SIV»). В этом случае устанавливается, что $i = i'$ и получаем уравнение зависимости:

$$i = \frac{c_2 - c_1}{2a_1}.$$

Тесты MIV. Для линейных индексных выражений массива, содержащих составные индексные переменные, используется тест Banerjee-НОД, расширенный для треугольных гнезд циклов.

Кроме этих тестов, используются различные методики, например: разбиение *сцепленных* ссылок массивов; символический тест и др. Отметим, что в работе [10] представлен более упрощенный и быстрый вариант Дельта-теста, называемый Эпсилон-тестом. В Эпсилон-тесте рассмотрены только самые простые случаи индексных выражений SIV и не обращается внимание на символические константы в выражениях, не используются *сцепленные* MIV-формы и НОД-тест, а также не рассматриваются треугольные границы цикла при использовании теста Банержи.

Форма коэффициентов: целочисленная константа
Форма границ цикла: целочисленная константа
Использование ограничений: да
Точность: точный для ZIV- и SIV-форм, для MIV иногда точен
Дистанционные вектора: иногда
Одновременное решение: да
Сложность: линейный для ZIV и SIV, полиномиальный или экспоненциальный для MIV
Уместность использования: недорогой для ZIV и SIV форм.

3.2.2. Точные тесты

3.2.2.1. Power-тест. Power-тест [8] представляет собой комбинацию обобщенного НОД-теста с уточненными ограничениями и метода исключения переменных Фурье—Моцкина. Тест применим во многих случаях, в которых информация является неизвестной во время компиляции, в частности, для тех случаев, которые включают неизвестные границы циклов.

Power-тест сначала пробует решить систему уравнений зависимости (3) с ограничениями (4) с помощью обобщенного НОД-теста (разд. 3.2.1.1). Если обобщенный НОД-тест не смог разрушить потенциальную зависимость, то это означает, что система (3) имеет целочисленные решения, но вопрос, находятся ли эти целочисленные решения внутри ограничений (4), остается открытым. В этом случае получается целочисленный **t** вектор, в котором значения $\mathbf{t}_m$, …, $\mathbf{t}_n$ являются неопределенными. Далее Power-тест пытается вычислить решения системы и найти значения дистанционного вектора зависимости, по уравнению $\mathbf{h} = \mathbf{tU}$, где $\mathbf{h}(x_1, x_2, \ldots, x_n)$ — вектор переменных системы и **U** — унимодулярная матрица. Если значения дистан-

ционного вектора определены и являются константами, то зависимость разрушается легко [8].

Если и этот метод не дал решений, то далее с помощью границ циклов и векторов направлений зависимости определяются верхние и нижние пределы $\mathbf{t}_m,\ldots,\mathbf{t}_n$. Полученное неравенство Power-тест решает с помощью метода исключения переменных Фурье—Моцкина.

Так как Power-тест использует метод исключения переменных Фурье—Моцкина, когда обобщенный НОД-тест неспособен разрушить зависимость, временная сложность теста в худшем случае теоретически показывает экспоненциальное время. В данном методе не предпринимаются никакие специальные действия, когда реализуется неточное проектирование, за исключением пометки результата как возможного консервативного.

Пример 8.

```
      do i1=1,50
            do i2=i1+1,50
S1:               X(i1,i2)=...
S2:               ...=X(i2,i1)...
            enddo
      enddo
```

Система уравнений зависимости имеет вид:

$$i_1 - j_2 = 0$$
$$i_2 - j_1 = 0,$$

где $1 \le i_1,\ j_1 \le 50,\ i_1+1 \le i_2,\ j_2 \le 50$.

В матричном виде $\boldsymbol{hA = c}$:

$$\left(i_1, j_1, i_2, j_2\right) \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & -1 \\ -1 & 0 \end{pmatrix} = (0,0).$$

Обобщенный НОД-тест вычисляет **UD** матрицу:

$$\left(\begin{array}{cccc|cc} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{array}\right).$$

Решая уравнение **tD = c,** получаем $\mathbf{t_1} = 0$, $\mathbf{t_2} = 0$, а уравнение **h = tU** дает значения:

$$i_1 = t_3,\ j_1 = t_4,\ i_2 = t_4,\ j_2 = t_3.$$

Расстояния зависимости не являются константами, поэтому с помощью границ индексных переменных создаем неравенство. Например, согласно нижней границе i_1, имеем неравенство $i_1 \geq 1$, отсюда получаем $t_3 \geq 1$, а для $j_1 \geq 1$ получаем $t_4 \geq 1$.

Нижние границы i_2 и j_2 дают следующие неравенства:

$$t_4 \geq t_3+1 \text{ и } t_3 \geq t_4+1.$$

После добавления таким же образом выражений для верхних границ и после некоторых нормализаций и устранений избыточных неравенств, получаем следующую систему неравенств:

$$1 \leq t_3 \leq 100$$
$$1 \leq t_4 \leq 100$$
$$t_3 + 1 \leq t_4 \leq t_3 - 1.$$

Сравнивая каждую нижнюю границу с каждой верхней границей t_4, в конечном счете, получаем:

$$t_3 + 1 \leq t_4 \leq t_3 - 1,$$

отсюда следует неравенство:

$$t_3 + 1 \leq t_3 - 1, \text{ или } 1 \leq -1,$$

которое является некорректным, таким образом, эта система уравнений и ограничений границ циклов не имеет решений, и операторы S1 и S2 не имеют зависимости по данным между собой.

Форма коэффициентов: линейные выражения
Форма границ циклов: линейные выражения

Использование ограничений: да, любая информация программы
Точность: точный
Дистанционные вектора: да
Одновременное решение: да
Сложность: экспоненциальная, но обычно полиномиальная
Уместность использования: дорогой.

3.2.2.2. Омега-тест. Омега-тест [9] совершенствует метод Фурье—Моцкина (разд. 3.1.3) до *целочисленных* решений. Он всегда возвращает точный ответ («**да/нет»)** и имеет в худшем случае экспоненциальную временную сложность.

Омега-тест определяет, существует или нет целочисленное решение произвольного множества линейных равенств и неравенств. Входом для теста служит множество линейных равенств $\sum_{1\le i\le n} a_i x_i = c$, представляемых в виде $\sum_{1\le i\le n} a_i x_i = 0$, и линейных неравенств $\sum_{1\le i\le n} a_i x_i \ge c$, представляемых в виде $\sum_{1\le i\le n} a_i x_i \ge 0$.

При описании теста будем полагаться, что все ограничения нормализованы в том смысле, что коэффициенты — целые числа и наибольший делитель коэффициентов (за исключением a_0) равен 1.

Следующим шагом является исключение ограничений в виде равенств. Для этого вводятся новые переменные. Далее проверяется наличие противоречащих неравенств; если таковые имеются, система ограничений противоречива и не имеет решений. Затем устраняются пары тесных неравенств и избыточные неравенства.

Как отмечено выше, в Омега-тесте метод Фурье—Моцкина был расширен до метода целочисленного программирования (разд. 3.1.3). Даже если $aL \le bU$, то может не быть целочисленного решения для x, где $aL \le abx \le bU$. Однако, если $aL + (a-1)(b-1) \le bU$, то целочисленное решение для x должно существовать. Это *темная* тень данной пары ограничений [9]. Темная тень есть пессимистическая аппроксимация целочисленной тени множества ограничений. Заметим, что если $a = 1$ или $b = 1$, то темная тень и вещественная идентичны, следовательно, они идентичны с целочисленной тенью.

Имеются случаи, когда действительная тень содержит целочисленные точки, а темная — нет. Тогда определение существования целочисленных решений исходного множества ограничений требует использования специальной техники [9]. Омега-тест начинает полный перебор пространства ре-

шений, рекурсивно генерируя и решая задачу целочисленного программирования, пока целочисленные решения или найдены, или опровергнуты.

При проектировании вдоль переменной x пара неравенств, оценивающая x сверху и снизу, порождает одно новое неравенство, не содержащее x. Совокупность таких неравенств определяет тень исходного объекта. Так как тень, получаемая этим путем, описывает вещественные решения, можно говорить о *вещественной тени* множества ограничений. Если не существует целочисленных точек в вещественной тени множества ограничений, то таких точек не существует и в исходном объекте.

Процесс решения систем равенств и неравенств в Омега-тесте аналогичен методу Фурье—Моцкина, за исключением того, что вычисляются две тени. Сравнивая верхний и нижний пределы в вычислении *вещественной* тени, получаем неравенство:

$$bL \le abx \le aU \;\Rightarrow\; bL \le aU.$$

Соответственно *темная тень* неравенства получается добавлением дополнительного ограничения к новому неравенству:

$$bL + (a - 1)\,(b - 1) \le aU.$$

Рассмотрим *пример* 3, использованный в разд. 3.1.3.
Дано:

$$\begin{aligned} 34 &\le x + 4y + 7z \le 34 \\ 0 &\le x \le 15 \\ -5 &\le y \le 10 \\ 3 &\le z \le 12. \end{aligned}$$

Выражая неравенства в терминах x, получаем:

$$\begin{aligned} 34 - 4y - 7z \le x &\le 34 - 4y - 7z \\ 0 \le x &\le 15 \\ -5 \le y &\le 10 \\ 3 \le z &\le 12. \end{aligned}$$

Сравнение верхних и нижних границ в терминах x дает те же самые результаты, как в методе Фурье—Моцкина, потому что коэффициент x равен 1 и $(1 - 1)\,(1 - 1) = 0$:

Верхняя граница	*Нижняя граница*	*Комбинация*	*Результат*
$34 - 4y - 7z \le x$	$x \le 34 - 4y - 7z$	$34 - 4y - 7z(+0) \le x \le 34 - 4y - 7z$	$0 \le 0$
$34 - 4y - 7z \le x$	$x \le 15$	$34 - 4y - 7z(+0) \le x \le 15$	$19 \le 4y + 7z$
$0 \le x$	$x \le 34 - 4y - 7z$	$0 (+0) \le x \le 34 - 4y - 7z$	$4y + 7z \le 34$
$0 \le x$	$x \le 15$	$0(+0) \le x \le 15$	$0 \le 15$

Затем, исключая y, получим следующую систему:

$$-7z + 19 \le 4y \le -7z + 34$$
$$-5 \le y \le 10$$
$$3 \le z \le 12$$

Сравним границы y. *Темная тень* неравенства в первой строке: $(9 = (4 - 1)(4 - 1))$.

Верхняя граница	*Нижняя граница*	*Комбинация*	*Результат*
$-7z + 19 \le 4y$	$4y \le -7z + 34$	$-28z + 76(+9) \le 16y \le -28z + 136$	$85 \le 136$
$-7z + 19 \le 4y$	$y \le 10$	$-7z + 19(+0) \le 4y < 40$	$-21 \le 7z$
$-5 \le y$	$4y \le -7z + 34$	$-20(+0) \le 4y \le -7z + 34$	$7z \le 54$
$-5 \le y$	$y \le 10$	$-5(+0) \le y \le 10$	$0 \le 15$

Нормализуя получившееся неравенство $-21 \le 7z \le 54$, получаем $-3 \le z \le 7$:

$$-3 \le z \le 7$$
$$3 \le z \le 12.$$

Затем исключается z. Темное дополнительное неравенство: $0 = (1 - 1)(1 - 1)$:

Верхняя граница	*Нижняя граница*	*Комбинация*	*Результат*
$-3 \le z$	$z \le 7$	$-3(+0) \le z \le 7$	$-3 \le 7$
$-3 \le z$	$z \le 12$	$-3 (+0) \le z \le 12$	$-5 \le 0$
$3 \le z$	$z \le 7$	$3 (+0) \le 7$	$0 \le 4$
$3 \le z$	$z \le 12$	$3 (+0) \le z \le 12$	$3 \le 12$

Конечное неравенство: $-15 \le 0 \le 4$. Так как исключены все переменные и не достигнуто никаких противоречий, тест делает вывод, что темная тень не пуста, и сообщает, что целочисленные решения существуют.

Если в течение вычисления вещественной тени получается противоречие неравенств, то Омега-тест сообщает, что вещественного решения не существует. Однако если бы противоречие возникло в результате неравенств темных теней, тогда тест обратился бы к стандартному методу Фурье—Моцкина, чтобы проверить, могли ли реальные решения быть исклю-

чены. Если они не могли бы быть исключены, Омега-тест начал бы рекурсивный поиск, как отмечено выше.

Форма коэффициентов: линейные выражения
Форма границ циклов: линейные выражения
Использование ограничений: да, любая информация программы
Точность: точный
Дистанционные вектора: да
Одновременное решение: да
Сложность: экспоненциальная, но обычно полиномиальная
Уместность использования: дорогой.

3.2.2.3. 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 делается вывод, что линейное уравнение не имеет целочисленного решения [11].

Рассмотрим случай уравнения зависимости с двумя переменными.

Процедура теста в случае уравнения с двумя переменными приведена ниже.

$$a_1x_1 + a_2x_2 = a_0 \qquad (7)$$

при условии

$$l_1 \le x_1 \le u_1 \text{ и } l_2 \le x_2 \le u_2, \qquad (8)$$

где a_0, a_1, a_2 являются целочисленными константами, и x_1, x_2 — целочисленные переменные.

Напомним, что линейную функцию двух переменных можно представить как прямую в двухмерном пространстве. 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, прямая проектируется на ось x_1. Эффективным интервалом решений x_1 является пересечение интервала прямой, 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.

Проектируемый интервал прямой $a_1x_1 + a_2x_2 = a_0$ на ось x_1 является:

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

Подобно тесту Банержи вычисляется предельное значение x_1 (разд. 3.1.2):

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

Пусть

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

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

означают верхнюю и нижнюю границу 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, если уравнение (7) 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 (7) с ограничениями (8) не имеет целочисленных решений. Иначе, так как интервал решений x_1 был сокращен (рис. 1), интервал решений x_2 изменится соответственно.

Аналогично, как описано выше, прямая проектируется на ось x_2. Пусть сокращенный интервал решений x_2 будет [$\lceil l_2^{'(1)} \rceil : \lfloor u_2^{'(1)} \rfloor$]. Если теперь $\lceil l_2^{'(1)} \rceil > \lfloor u_2^{'(1)} \rfloor$, то уравнение (7) с ограничениями (8) не имеет целочисленных решений. Иначе линия снова проектируется на ось x_1 с целью сокращения интервала решений x_1.

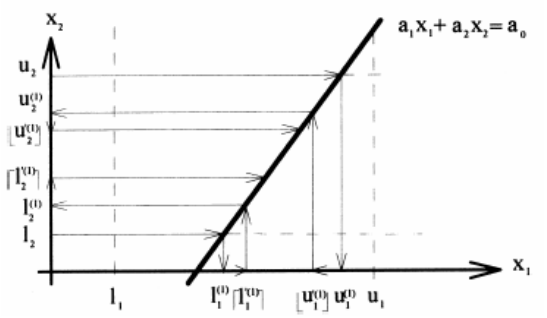

Рис. 1. Сокращение интервала решения

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 (7) при условии ограничения (8) имеются следующие случаи, где IR-тест принимает соответствующие решения.

1. Если интервалы решений x_1 или x_2 были преобразованы 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 (7) с ограничением (8) 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 (7) с ограничением (8) содержит, по крайней мере, одно целочисленное решение.
   (a) Если оба интервала решений свелись 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 (7) с ограничением (8) имеет только одно целочисленное решение: $(x_1, x_2)=(\lceil l_1'^{(m)} \rceil, \lceil l_2'^{(m)} \rceil)$.
   (b) Иначе, уравнение (7) с ограничением (8) 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 уменьшают, добавляя и вычитая единицу к нижней и

верхней границе 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 [11].

Пример 8.

```
      do i=1, 10
S1:           C(3i – 10)=...
S2:           ...=C(2i + 17)...
      enddo
```

Уравнение зависимости имеет вид: $3x_1 - 2x_2 = 27$, где $1 \le x_1, x_2 \le 10$.

Проектируя прямую, описанную данным уравнением, на оси x_1 получаем $x_1 = (2/3)\, x_2 + (27/3)$ где $1 \le x_1 \le 10$, $1 \le x_2 \le 10$.

Верхняя и нижняя границы соответственно равняются:

$$l_1^{(1)} = (2/3)^+(1) - (2/3)^-(10) + (27/3) = (29/3),$$
$$u_1^{(1)} = (2/3)^+(10) - (2/3)^-(1) + (27/3) = (47/3).$$

Эффективным интервалом решений x_1 становится

$$[(29/3):(47/3)] \cap [1:10] = [29/3:10].$$

Отсюда, $x_1 \in [\lceil 29/3 \rceil : \lfloor 10 \rfloor] = [10:10]$, т.е. $l_1'^{(1)} = 10$, $u_1'^{(1)} = 10$.

Так как этот интервал не пуст, спроектируем прямую $3x_1 - 2x_2 = 27$ на интервале $10 \le x_1 \le 10$ на оси x_2. Эффективным интервалом решений по x_2 является [3/2:3/2]. Так как $\lceil 3/2 \rceil > \lfloor 3/2 \rfloor$, то уравнение не имеет целочисленных решений в данном интервале, и операторы **S1** и **S2** не имеют зависимости по данным.

Отметим, что в данном примере НОД-тест, тест Банержи и I-тест показывают зависимости по данным. IR-тест зависит от границ циклов, т.е. он

просто неприменим, когда значения границ циклов неизвестны или не являются константами.

В [26] IR-тест расширен для анализа зависимостей по данным с учетом векторов направлений. Многомерный вариант теста представлен в работе [27].

Форма коэффициентов: целочисленная константа
Форма границ циклов: целочисленная константа
Использование ограничений: да
Точность: точный
Дистанционные вектора: нет
Одновременное решение: нет
Сложность: полиномиальная
Уместность использования: обычно дорогой.

3.3. Другие тесты на зависимость по данным

Как отмечено, в анализе зависимостей по данным используются различные математические методы. CM-тест (Constraint-Matrix test) описан в [16]; он основан на использовании *симплекс-метода*, модифицированного для решения задач целочисленного программирования. В общем случае CM-тест может зацикливаться. Неясно также, как эффективно он работает на практике. В работе [28] показан, что целочисленный симплекс-метод работает медленнее, но точнее чем метод Фурье–Моцкина.

Алгоритм Шостака [17] основан на использовании *теоретико–графового* метода для решения систем линейных неравенств. В [29] описан модифицированный вариант алгоритма Шостака.

В системе SUIF Стенфордского университета анализ зависимостей по данным базируется на алгоритме Майдана [12] (MHL-тест) и состоит из серии точных тестов, каждый из которых применим в ограниченной области. Его последний тест — алгоритм исключения Фурье–Моцкина, расширенный для решения целочисленных задач. Анализатор системы также способен обрабатывать некоторые простые нелинейные доступы к массивам.

K-тест [30] тоже состоит из библиотеки тестов на зависимость, но в отличие от MHL-теста, вместо конкретной стратегии применения тестов используются методы искусственного интеллекта.

4. ЗАКЛЮЧЕНИЕ

Как следует из приведенного обзора, проблема выявления зависимостей по данным весьма актуальна при распараллеливании последовательных программ. В работе рассмотрены основные тесты на зависимость и даны сопоставления их сильных и слабых сторон как на примерах, так и по оцениваемым характеристикам отдельных критериев. Приведена обширная классификация тестов на зависимость по использованию различных математических инструментов и по сложности применения их на практике. На рис. 2 приведена общая схема классификации тестов. Стрелками показаны связи между тестами.

Рассмотренная классификация алгоритмов зависимости по данным имеет практический интерес. Она дает возможность выработать оптимальную стратегию применений тестов на зависимость при распараллеливании последовательных программ. На практике алгоритм анализа зависимостей по данным должен быть по мере возможности «наиболее точным» и более «дешевым».

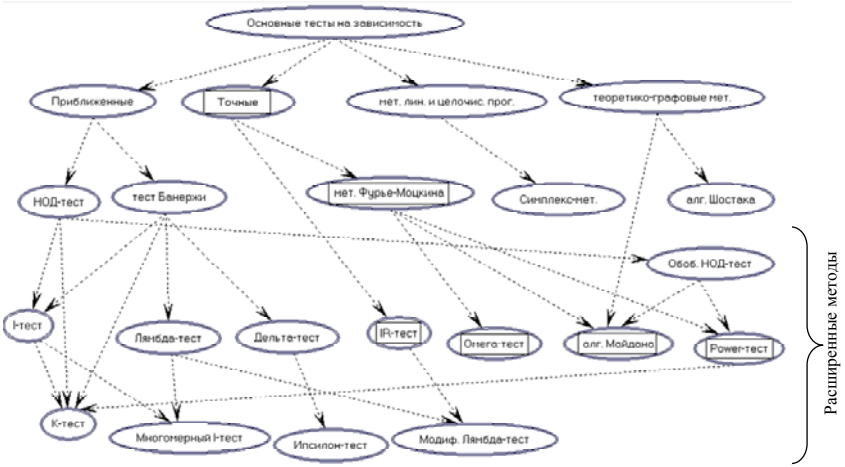

Рис. 2. Схема классификация тестов на зависимость.
Прямоугольниками выделены точные тесты

СПИСОК ЛИТЕРАТУРЫ

1. **Евстигнеев В.А.** Анализ зависимостей: состояние проблемы // Системная информатика: Сб. науч. тр. / Ин-т систем информатики СО РАН. — Новосибирск: Наука, 2000. — Вып. 7. — С. 112–173.
2. **Banerjee U.** Data dependence in ordinary programs. — Urbana, 1976. — (Tech. Rep. / Univ.III.; 76-837).
3. **Banerjee U.** An introduction to a formal theory of dependence analysis // The J. of Supercomputing. — 1988. — Vol.2. — P. 133–149.
4. **Banerjee U.** Dependence Analysis . Boston, Mass.: Kluwer Academic Publishers. — 1997. — P. 106.
5. **Kong X., Klappholz D., 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.
6. **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, N 1. — P. 26–34.
7. **Goff G., Kennedy K., Tseng C.** Practical Dependence Testing // Proc. of the ACM SIGPLAN 91 Conference on Programming Language Design and Implementation, June 1991. — P. 15–29.
8. **Wolfe M., Tseng C.** The power test for data dependence // IEEE Trans. Parallel Distrib. Syst. — 1992. — Vol. 3, N 5. — P. 591–601.
9. **Pugh W.** The Omega test: a fast and practical integer programming algorithm for dependence analysis // Communs. of the ACM. — 1992. — Vol. 35, N 8 . — P. 102–114.
10. **Pugh W., Speismean T**. On Fast Array Data Dependence Tests. — Univ. of Maryland, College Park, January 3, 1999.
11. **Huang T.-C., Yang C.-M.** Data dependence analysis for array references // J. Systems and Software. — 2000. — Vol. 52. — P. 55–65.
12. **Maydan D., Hennessy J., Lam M.** Efficient and Exact Data Dependence Analysis // Proc. of the SIGPLAN Conf. on Programming Language Design and Implementation, June 1991. — P. 1–14.
13. **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, N 3. — P. 356–364.
14. **Eisenbeis C., Sogno J.-C.** A general algorithm for data dependence analysis // Proc. of the Sixth ACM International Conference on Supercomputing, Washington, DC, July 1992.
15. **Dantzig G., Eaves B.** Fourier-Motzkin Elimination and its Dual // J. of Combinatorial Theory. A. — 1973. — Vol. 14. — P. 288–297.
16. **Wallace, D. R.** Dependence of multi-dimensional array references // Proc. of 1988 Internat. Conf. on Supercomputing (Saint-Malo, France, July 1988).

17. **Shostak R.** Deciding linear inequalities by computing loop residues // ACM J. — 1981. — Vol. 28, N 4. — P. 769–779.
18. **Psarris K.** On exact data dependence analysis // Proc of the Sixth ACM Internat. Conf. on Supercomputing, Washington, DC, July 1992.
19. **Psarris K., Klappholz D., Kong X.** On the accuracy of the Banerjee test, shared memory multiprocessors // J. Parallel Distrib. Comput. — 1991. — Vol.12 , N 2.
20. **Psarris K**. Program analysis techniques for transforming programs for parallel execution // Parallel Computing. — 2002. — Vol. 28. — P. 455–469.
21. **Chang W.-L., Chu C.-P., Wu J.** The generalized Lambda test // Proc. of the First Merged Symp. on IPPS/SPDP, Orlando, FL, March 1998. — P. 181–186.
22. **Chang W.-L., Chu C.-P.** The infinity Lambda test: A multi-dimensional version of Banerjee infinity test // Parallel Computing. — 2000. — Vol. 26. — P. 1275–1295.
23. **Psarris K., Kong X., Klappholz D.** The direction vector I test // IEEE Transactions on Parallel and Distributed Systems. — 1993. — Vol. 4, N 11. — P. 1280–1290.
24. **Chang, W.-L., Chu, C.-P., Wu, J.** A multi-dimensional version of the I test // Parallel Computing. — 2001. — Vol. 27. — P. 1783–1799.
25. **Chang, W.-L., Chu, C.-P., Wu, J.** A precise dependence analysis for multi-dimensional arrays under specific dependence direction // J. of Systems and Software. — 2002. — Vol. 63. — P. 99–112.
26. **Huang T.-C., and Yang C.-M**. Data dependence analysis with direction vector for array references // Computers and Electrical Engineering. — 2001. — Vol. 27. — P. 375–393.
27. **Арапбаев Р.Н., Осмонов Р.А.** Анализа зависимостей по данным для многомерных массивов на базе модифицированного λ-теста // Проблемы интеллектуализации качества систем информатики. — Новосибирск, ИСИ СО РАН, 2006. — С. 7–23.
28. **Дроздов А.Ю., Корнев Р.М., Боханко А.С.** Индексный анализ зависимостей по данным // Информационные технологии и вычислительные системы. — 2004. — Т. 3. — С. 27–37.
29. **Yang C.-T., Tseng S.-S., Shih W.-C.** The K Test: an Exact and Efficient Knowledge-based Data Dependence Testing Method for Parallelizing Compiler // Proc. Natl. Sci. Counc. ROC(A). — 2000. — Vol. 24, N. 5. — P. 362–372.
30. **Логачева С.А.** Анализ зависимостей по данным на базе алгоритма Шостака // Поддержка супервычислений и Интернет–ориентированные технологии. — Новосибирск, ИСИ СО РАН, 2001. — С. 31–43.

Р.Н. Арапбаев, В.А. Евстигнеев, Р.А. Осмонов

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ТЕСТОВ НА ЗАВИСИМОСТЬ ПО ДАННЫМ

Препринт
141

Рукопись поступила в редакцию 11.12.06
Рецензент П.Г. Емельянов
Редактор З.В. Скок

Подписано в печать 28.12.06
Формат бумаги 60 × 84 1/16 Объем 2.2 уч.-изд.л., 2.4 п.л.
Тираж 60 экз.

Центр оперативной печати «Оригинал 2», г. Бердск, 49-а, оф. 7, тел./факс 8 (241) 5 38 77