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

В. А. Непомнящий, Е. А. Мацко

ПРИМЕНЕНИЕ СИМВОЛИЧЕСКОГО МЕТОДА ЭЛИМИНАЦИИ ИНВАРИАНТОВ ЦИКЛОВ К ВЕРИФИКАЦИИ ПРОГРАММ ЛИНЕЙНОЙ АЛГЕБРЫ

Препринт
102

Новосибирск 2003

Описана методология верификации программ линейной алгебры, которая включает символический метод элиминации инвариантов циклов, адаптированный к программам линейной алгебры, правила вывода условий корректности, не требующие инвариантов циклов, а также базу знаний, состоящую из операций над массивами и понятий линейной алгебры. Эта методология иллюстрируется примерами верификации программ обращения матрицы и решения системы линейных уравнений методом Гаусса.

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

V. A. Nepomniaschy, E. A. Matsko

APPLYING THE SYMBOLIC METHOD FOR LOOP INVARIANT ELIMINATION TO LINEAR ALGEBRA PROGRAM VERIFICATION

Preprint
102

Novosibirsk 2003

A methodology for linear algebra program verification is described. It includes the symbolic loop invariant elimination method adapted to linear algebra programs, proof rules without loop invariants and a knowledge base consisting of linear algebra notions and operations over arrays. This methodology is illustrated by examples of verification of two programs: inversion of a matrix and decision of a linear equation system by the Gauss method.

1. ВВЕДЕНИЕ

Аксиоматический подход к верификации программ базируется на методе Хоара и состоит из следующих этапов: аннотирования программ посредством пред- и постусловий, а также инвариантов циклов, вывода условий корректности и их доказательства [13]. Аннотирование циклов остается трудной проблемой, которая является значительным препятствием на пути к практической верификации программ [15].

Естественный подход к упрощению верификации — использование финитных итераций, аналогичных for-циклам, для которых обеспечена терминация. Общая форма финитной итерации как итерации над всеми элементами произвольной структуры данных была предложена в [16], где мотивировалось распространение таких циклов в практическом программировании.

Символический метод верификации for-циклов, телом которых является оператор присваивания элементам массива, был развит в [4, 5, 6, 7, 9]. Этот метод базируется на использовании операции замены, которая в символической форме представляет действие for-цикла и позволяет выразить его инвариант. Метод использует специальную технику доказательства условий корректности, включающих операцию замены. В [10, 11] символический метод был распространен на финитные итерации общего вида без ограничений на тела циклов.

Цель настоящей работы состоит в том, чтобы описать новую методологию применения символического метода к верификации программ линейной алгебры. Для простых программ линейной алгебры такая методология представлена в [4, 6, 7]. Однако для верификации используемых на практике оптимальных программ линейной алгебры эту методологию необходимо существенно расширить.

Символический метод элиминации инвариантов циклов, адаптированный к программам линейной алгебры, описан в разд. 2. База знаний, состоящая из операций над массивами и понятий линейной алгебры, представлена в разд. 3 и 4. Верификация программы обращения матрицы методом Гаусса описана в разд. 5. Верификация программы решения системы линейных уравнений методом Гаусса представлена в разд. 6. В заключении обсуждаются перспективы применения символического метода к верификации программ линейной алгебры.

Эта работа частично поддержана грантом Российского фонда фундаментальных исследований № 00-01-00909.

2. СИМВОЛИЧЕСКИЙ МЕТОД ЭЛИМИНАЦИИ ИНВАРИАНТОВ ЦИКЛОВ

2.1. Финитные итерации над структурами данных

Введем следующие обозначения. Пусть $\{s_1, \ldots, s_n\}$ обозначает мультимножество, состоящее из элементов $s_1, \ldots, s_n, U_1 - U_2$ — разность мультимножеств U_1 и U_2, $U_1 \bigcup U_2$ — их объединение, а $\mid U \mid$ — мощность конечного мультимножества U. Пусть $[v_1, \ldots, v_m]$ — вектор, состоящий из элементов v_i ($\emptyset$ — "пустой" вектор). Операция конкатенации $con(V_1, V_2)$ векторов V_1 и V_2 определяется обычным образом.

Напомним понятие структуры данных [16]. Пусть $memb(S)$ — мультимножество элементов структуры S, $empty(S)$ — предикат "$memb(S)$ — пусто", $choo(S)$ — функция, значение которой — элемент $memb(S)$, $rest(S)$ — функция, значение которой — структура S' такая, что $memb(S') = memb(S) - \{choo(S)\}$. Функции $choo(S)$ и $rest(S)$ неопределены тогда и только тогда, когда $empty(S)$. Таким образом, это определение абстрагируется от способа задания функций $choo(S)$ и $rest(S)$, что придает ему значительную гибкость.

Напомним определение нескольких полезных функций [10, 11], связанных со структурой S. Пусть в случае $\neg empty(S)$ и $memb(S) = \{s_1, \ldots, s_n\}$ $vec(S) = [s_1, \ldots, s_n]$, где $s_i = choo(rest^{i-1}(S))$ для $i = 1, \ldots, n$. При $empty(S)$ положим $vec(S) = \emptyset$. Функция $vec(S)$ задает развертку структуры S, которая однозначно определяет ее использование. Две структуры S_1 и S_2 называются эквивалентными ($S_1 = S_2$) тогда и только тогда, когда $vec(S_1) = vec(S_2)$. Значением функции $head(S)$ является такая структура, что $vec(head(S)) = [s_1, \ldots, s_{n-1}]$ при $vec(S) = [s_1, \ldots, s_n]$ и $n \geq 2$. Если $n = 1$, то выполняется $empty(head(S))$. Если $vec(S) = [s_1, \ldots, s_n]$, то значением функции $last(S)$ будет элемент s_n. Пусть функции $head(S)$ и $last(S)$ неопределены в случае $empty(S)$. Пусть $str(s)$ означает структуру, содержащую единственный элемент s. Операция конкатенации структур $con(S_1, S_2)$ определена в [10] таким образом, что $vec(con(S_1, S_2)) = con(vec(S_1), vec(S_2))$. Пусть $con(S, s) = con(S, str(s))$ и $con(s, S) = con(str(s), S)$, а $con(S_1, S_2, S_3) = con(con(S_1, S_2), S_3)$. Легко доказать, что при $\neg empty(S)$ справедливо $con(choo(S), rest(S)) = con(head(S), last(S)) = S$, а при $\neg empty(rest(S))$ выполняется $head(rest(S)) = rest(head(S))$.

Рассмотрим финитную итерацию вида

$$\textbf{for } x \textbf{ in } S \textbf{ do } v := body\,(v, x) \textbf{ end}, \tag{1}$$

где S — структура, возможно иерархическая, x — переменная, называемая параметром итерации, v — вектор переменных, не содержащий x, причем тело итерации $v := body(v, x)$ не изменяет структуры S. Пусть v_0 — начальное значение вектора v. При $empty(S)$ результатом итерации считается v_0. Если $\neg empty(S)$ и $vec(S) = [s_1, \ldots, s_n]$, то тело цикла $v := body(v, x)$ выполняется последовательно для x, принимающего значения $s_1, \ldots, s_n$.

2.2. Вывод условий корректности без инвариантов и их доказательство

Пусть $R(y \leftarrow exp)$ означает результат подстановки выражения exp вместо всех вхождений переменной y в формулу R. Пусть $R(vec \leftarrow vexp)$ означает результат одновременной подстановки в формулу R компонент вектора выражений $vexp$ вместо всех вхождений соответстующих компонент вектора vec. Правило вывода условий корректности $rl1$ [11] для итерации (1) использует операцию замены $rep(v, S, body)$, где $body$ означает функцию, ассоциированную с правой частью тела итерации (1). Операция замены $rep(v, S, body)$ представляет действие итерации (1) и определяется следующим образом. Пусть $rep(v, S, body) = v_n$, где $v_0 = v$, $n = 0$ при $empty(S)$, а $v_i = body(v_{i-1}, s_i)$ для каждого $i = 1, \ldots, n$ при $\neg empty(S)$ и $vec(S) = [s_1, \ldots, s_n]$. Следующие результаты [11] описывают полезные свойства операции замены.

Теорема 1.
1.1. Итерация (1) эквивалентна оператору векторного присваивания $v := rep(v, S, body)$.
1.2. $rep(v,\ con(S_1, S_2),\ body) = rep(rep(v, S_1, body),\ S_2,\ body)$.
1.3. $rep(v, str(s), body) = body(v, s)$.

Следствие 1.
Пусть $\neg empty(S)$. Тогда справедливо
1.1. $rep(v, S, body) = body(rep(v,\ head(S),\ body),\ last(S))$.
1.2. $rep(v, S, body) = rep(body(v,\ choo(S)),\ rest(S),\ body)$.

Из теоремы 1.1 вытекает следующее правило вывода условий корректности:

rl1. $\{P\}\ prog\ \{Q(v \leftarrow rep(v, S, body))\} \vdash$
$\{P\}\ prg;\ \textbf{for}\ x\ \textbf{in}\ S\ \textbf{do}\ v := body(v, x)\ \textbf{end}\ \{Q\}$,

где P — предусловие, Q — постусловие, не зависящее от параметра ите-

рации x, prg — фрагмент программы, а $\{P\}\ prg\ \{Q\}$ означает частичную корректность программы prg относительно P и Q.

В результате применения правила rl1 порождаются условия корректности, содержащие операцию замены. Для доказательства этих условий применяется как универсальная техника, базирующаяся на принципах индукции, так и проблемно-ориентированная техника. Рассмотрим принцип индукции, который служит для доказательства свойств операции замены.

Пусть $prop(rep(v, S, body))$ обозначает свойство, которое выражается формулой логики первого порядка со свободной переменной S. Эта формула строится из операции замены $rep(v, S, body)$, функциональных символов, переменных и констант посредством булевских операций, кванторов 1-го порядка и подстановки констант вместо переменных из v.

Индукцией по параметру $k =\mid memb(S) \mid$ нетрудно установить следующий принцип.

Принцип индукции 1. Свойство $prop(rep(v,\ S,\ body))$ выполняется для произвольной структуры S, если существует константа $c \geq 0$ такая, что справедливы два условия:

(1) для любой структуры S такой, что $\mid memb(S) \mid \leq c$, выполняется свойство $prop(rep(v, S, body))$;

(2) для любой структуры S такой, что $\mid memb(S) \mid > c$, выполняется свойство $prop(rep(v, head(S), body)) \rightarrow prop(rep(v,\ S,\ body))$.

2.3. Преобразования массивов

Многие циклы в программах линейной алгебры осуществляют преобразования массивов, и их можно представить в виде итерации

$$\textbf{for } x \textbf{ in } S \textbf{ do } A[ind(x)] := exp(A, x) \textbf{ end}, \tag{2}$$

где A — массив произвольной размерности, $ind(x)$ — функция, не зависящая от массива A, exp — выражение, которое может зависеть от массива A и параметра цикла x.

Чтобы представить итерацию (2) в виде итерации (1), тело итерации (2) представим в виде $A := upd(A, ind(x), exp(A, x))$, где $upd(A, i, e)$ означает массив, полученный из массива A, если в качестве элемента с индексом i взять значение выражения e [14]. Тогда по теореме 1.1 итерация (2) эквивалентна оператору $A := rep\,(A, S, upd(A, ind, exp))$.

Наряду с оператором (2) последовательного преобразования массивов рассмотрим оператор параллельного преобразования массивов

$$\textbf{forpar } x \textbf{ in } S \textbf{ do } A := upd(A, ind(x), exp(A, x)) \textbf{ end}. \qquad (3)$$

При выполнении оператора (3) вначале вычисляются функция $ind(x)$ и выражение $exp(A, x)$ для всех $x \in memb(S)$, а затем производится одновременная замена всех элементов массива A с индексами $ind(x)$ значениями соответствующих выражений $exp(A, x)$. Эта семантика требует уточнения в случае, когда функция $ind(x)$ не является однозначной. Областью замены называется множество $ind(S)$, состоящее из элементов вида $ind(s)$ для $s \in memb(S)$. Пусть $vec(S) = [s_1, ..., s_n]$. Определим функцию $moc(S, k)$, обратную к функции $ind(x)$, следующим образом. При $k \in ind(S)$ положим $moc(S, k) = x$, если $k = ind(x)$ и существует число i $(1 \le i \le n)$ такое, что $x = s_i$, причем $k \ne ind(s_j)$ для любого j, где $i < j \le n$. В случае $k \notin ind(S)$ функция $moc(S, k)$ не определена. Выполнение оператора (3) состоит в одновременной замене элементов $A[k]$ значениями выражений $exp(A, moc(S, k))$ для всех $k \in ind(S)$. Определим операцию параллельной замены $reppar(A, S, upd(A, ind, exp)) = A_n$, где $A_0 = A, A_j = upd(A_{j-1}, ind(s_j), exp(A, s_j))(j = 1, ..., n)$.

Теорема 2. Оператор (3) эквивалентен оператору

$$A := reppar(A, S, upd(A, ind, exp)).$$

Доказательство. Применяется индукция по $|\,memb(S)\,|$. При $empty(S)$ теорема 2 очевидна. Пусть $\neg empty(S)$. Тогда оператор (3) эквивалентен программе $prog1$:

$A_0 := A$; **forpar** x **in** $head(S)$ **do** $A := upd(A, ind(x), exp(A, x))$ **end**;
$A := upd(A, ind(last(S)), exp(A_0, last(S)))$.

Действительно, если $ind(last(S)) \notin ind(head(S))$, то это утверждение очевидно, а в противном случае оно следует из

$moc(S, ind(last(S))) = last(S)$.

По предположению индукции программа $prog1$ эквивалентна программе $prog2$:

$A_0 := A; A := reppar(A, head(S), upd(A, ind, exp))$;
$A := upd(A, ind(last(S)), exp(A_0, last(S)))$.

Ввиду $vec(head(S)) = [s_1, ..., s_{n-1}]$ и $last(S) = s_n$ программа $prog2$ представима в виде

$$A_0 := A; A := A_{n-1}; A := upd(A, ind(s_n), exp(A_0, s_n))$$

и, следовательно, эквивалентна оператору $A := A_n$.

Следствие 2.
$reppar(A, S, upd(A, ind, exp))[k] =$
if $k \in ind(S)$ **then** $exp(A, moc(S, k))$ **else** $A[k]$.

Из теоремы 2 вытекает также следующее правило вывода условий корректности

rl2. $\{P\}prog\{Q(A \leftarrow reppar(A, S, upd(A, ind, exp)))\} \vdash$
$\{P\}prog;$ **forpar** x **in** S **do** $A := upd(A, ind(x), exp(A, x))$ **end** $\{Q\}$,

где P — предусловие, Q — постусловие, не зависящее от x, $prog$ — фрагмент программы.

Пусть вхождение массива A в выражение $exp(A, x)$ имеет вид $A[r_i(x)](i = 1, ..., t)$. Обозначим через $pred(vec(S), j)$ множество $\{s_1, ..., s_{j-1}\}$ при $1 < j \leq n$. Пусть $pred(vec(S), 1)$ — пустое множество. Следующий результат [11] дает достаточное условие совпадения операций последовательной и параллельной замены, а также эквивалентности операторов (2) и (3).

Теорема 3.

$$rep(A, S, upd(A, ind, exp)) = reppar(A, S, upd(A, ind, exp)),$$

если

$$r_i(s_j) \notin ind(pred(vec(S), j))$$

для всех $i = 1, ..., t, j = 2, ..., n$.
В случае выполнения условия теоремы 3 для вычисления операции последовательной замены можно применить следствие 2.

С целью упрощения функции $ind(x)$ будет использоваться преобразование итерации (2) над структурой S к итерации над структурой $ind(S)$, где $vec(ind(S)) = [ind(s_1), ..., ind(s_n)]$ при $vec(S) = [s_1, ..., s_n]$, причем положим $empty(ind(S))$ в случае $empty(S)$. Заметим, что если $ind(x)$ — однозначная функция, то $moc(S, ind(x)) = x$ для каждого $x \in memb(S)$. Отсюда следует, что при однозначной функции $ind(x)$ итерация (2) эквивалентна итерации

$$\textbf{for } x \textbf{ in } ind(S) \textbf{ do } A[x] := exp(A, moc(S, x)) \textbf{ end}. \qquad (4)$$

3. АКСИОМАТИЗАЦИЯ ОПЕРАЦИЙ НАД МАССИВАМИ

В качестве языка спецификаций программ линейной алгебры используется логический язык, формулы которого строятся из понятий (функций и предикатов) с помощью логических операций первого порядка. Понятия языка спецификаций программ линейной алгебры можно разделить на три класса. Первый класс — это универсальные понятия (те, которые есть во всех языках спецификаций). К нему относятся стандартные арифметические и логические понятия. Второй класс, описанный в настоящем разделе содержит понятия, связанные со структурами данных, то есть с массивами для данной проблемной области. К третьему классу, описанному в разд. 4, относятся проблемно-ориентированные понятия, связанные с задачами линейной алгебры.

3.1. Индексные множества

Областью определения матрицы или какой-либо ее части будем считать индексные множества — конечные множества пар натуральных чисел. Произвольную матрицу $M[k : l, m : n]$ будем трактовать как отображение индексного множества $mat(k, l, m, n)$ в множество вещественных чисел. Определим семь базисных индексных множеств:

- $set(k, m) = \{(u, v)|(u = k \wedge v = m)\}$ — множество, состоящее из одного элемента (k, m);
- $row(l, m, n) = \{(u, v)|(u = l \wedge m \leq v \leq n)\}$ — множество, определяющее l-ю строку матрицы от m-го до n-го столбца;
- $col(m, k, l) = \{(u, v)|(v = m \wedge k \leq u \leq l)\}$ — множество, определяющее m-ый столбец матрицы от k-й до l-й строки;
- $dig(k, l, m) = \{(u, v)|(m - k + u = v \wedge k \leq u \leq l)\}$ — множество, определяющее диагональ матрицы от пересечения m -го столбца и k-й строки до l-й строки;
- $mat(k, l, n, m) = \{(u, v)|(k \leq u \leq l \wedge m \leq v \leq n)\}$ — множество, описывающее полосу матрицы от k-й до l-й строки и от m-го до n-го столбца;
- $ldmat(k, l, m, n) = \{(u, v)|(k \leq u \leq l \wedge m \leq v \leq n \wedge v \leq m - k + u)$ — множество, определяющее поддиагональную полосу матрицы до l-й строки и n-го столбца, в случае, когда диагональ начинается от пересечения k-й строки и m-го столбца;
- $rdmat(k, l, m, n) = \{(u, v)|(k \leq u \leq l \wedge m \leq v \leq n \wedge u \leq k - m + v)$ — множество, описывающее наддиагональную полосу матрицы до

l-й строки и n-го столбца, в случае, когда диагональ начинается от пересечения k-й строки и m-го столбца.

Заметим, что первые шесть индексных множеств были введены в [7]. Ограничимся классом индексных множеств, построенных из пустого (ε) и базисных индексных множеств с помощью операций объединения, пересечения и разности. Пустое индексное множество — область определения пустой матрицы Ω.

Приведем некоторые аксиомы, описывающие свойства индексных множеств.

AM.IND1 $\cup_{x=l}^{r} set(\alpha, x) = row(\alpha, l, r)$.

AM.IND2 $\cup_{x=l}^{r} set(x, \alpha) = col(\alpha, l, r)$.

AM.IND3 $\cup_{x=l}^{r} col(x, p, q) = mat(p, q, l, r)$.

AM.IND4 $\cup_{x=l}^{r} row(x, p, q) = mat(l, r, p, q)$.

3.2. Элементарные операции

Рассмотрим предикаты равенства матриц [7] и одномерных массивов. Равенство ($M_1 = M_2$) означает, что у матриц одинаковые области определения и отображения. Слабое равенство ($M_1 \approx M_2$) говорит о том, что матрицы имеют одинаковое количество строк и столбцов и при наложении матриц соответствующие компоненты совпадают. Более строгое определение равенства матриц дается аксиомой AM.EQ1, слабого равенства — аксиомой AM.EQ2 из [7]. Введем аналогичное понятие для одномерных массивов. Равенство двух массивов с одинаковыми границами означает совпадение соответствующих компонент массивов. Аксиома AM.EQ′ дает определение этого понятия.

AM.EQ′ $(\sigma[k:l] = \tau[k:l]) \equiv (\forall u \in \{k, \ldots, l\}(\sigma[u] = \tau[u]))$.

Рассмотрим операцию конкатенации матриц [7].

Операция конкатенации $con(M_1[k_1 : l_1, m_1 : n_1], M_2[k_2 : l_2, m_2 : n_2])$ определена только для матриц с одинаковым числом строк и означает приписывание к M_1 всех столбцов матрицы M_2. Это понятие описывают аксиомы AM.CON1–AM.CON3 из [7], а также следующие аксиомы:

AM.CON4
$k \le l \wedge m \le s \le n \Rightarrow M[k:l, m:n] = con(M[k:l, m:s], M[k:l, s+1:n])$.

AM.CON5 $con(M, \Omega) = con(\Omega, M) = M$.

Обозначим $con(con(M_1, M_2), M_3)$ через $con(M_1, M_2, M_3)$.

3.3. Операции замены

Рассмотрим операцию замены в матрице, которая обозначалась в [7] через *rep*. Для матрицы M, индексного множества S и выражения $e(s,t)$ через $UPD(M,S,e(s,t))$ обозначим матрицу, полученную из M, если для каждого элемента $(s,t) \in S$ в качестве (s,t)-го компонента взять значение выражения $e(s,t)$. Аксиомы AM.REP1–AM.REP4 из [7] дают формальное описание этого понятия. Для сокращения записи выражение

$$UPD(\ldots UPD(UPD(M_1,S_1,e_1),S_2,e_2),\ldots,S_n,e_n)$$

при $n>1$ будем иногда представлять в виде

$$UPD(M_1,S_1,e_1,S_2,e_2,\ldots,S_n,e_n).$$

AM.UPD1 $UPD(M,set(u,v),e(s,t)) = upd(M,(u,v),e(u,v))$.

AM.UPD2 $def(UPD(M,S,e(s,t))) = def(M)$.

AM.UPD3 $(u,v) \in S \cap def(M) \Rightarrow UPD(M,S,e(s,t))[u,v] = e(u,v)$.

AM.UPD4
$$(u,v) \notin S \wedge (u,v) \in def(M) \Rightarrow UPD(M,S,e(s,t))[u,v] = M[u,v].$$

Следующие теоремы взяты из [7]:

TM.UPD1
$$\forall (u,v) \in S \cap def(M)(M[u,v] = e(u,v)) \Rightarrow UPD(M,S,e(s,t)) = M.$$

TM.UPD2

$$(S_1 \cap S_2) \cap def(M) = \varepsilon \Rightarrow \begin{array}{l} UPD(UPD(M,S_1,e_1(s,t)),S_2,e_2(s,t)) = \\ UPD(UPD(M,S_2,e_2(s,t)),S_1,e_1(s,t)). \end{array}$$

TM.UPD3
$$\begin{array}{l} UPD(UPD(M,S_1,e_1(s,t)),S_2,e_2(s,t)) = \\ UPD(UPD(M,S_1 \cap S_2,e_1(s,t)),S_2,e_2(s,t)). \end{array}$$

TM.UPD4
$$UPD(UPD(M,S_1,e(s,t)),S_2,e(s,t)) = UPD(M,S_1 \cup S_2,e(s,t)).$$

TM.UPD5

$$\begin{array}{l} S_3 \cap def(M) \subseteq S_2 \wedge \forall (u,v) \in S_3 \cap def(M)(e_1(u,v)) = e_2(u,v) \Rightarrow \\ UPD(UPD(M,S_1,e_1(s,t)),S_2,e_2(s,t)) = \\ UPD(UPD(M,S_1 \cup S_3,e_1(s,t)),S_2 \setminus S_3,e_2(s,t)). \end{array}$$

TM.UPD6 $UPD(M, S, e(s,t)) = UPD(M, S \cap def(M), e(s,t))$.

TM.UPD7 $mat(k,l,m,n) \subseteq def(M) \Rightarrow$
$UPD(M, S, e(s,t))[k:l, m:n] = UPD(M[k:l, m:n], S, e(s,t))$.

Следующая теорема показывает связь операции UPD с операцией параллельной замены $reppar(A, S, upd(A, ind, exp))$, когда $ind(x)$ есть тождественная функция I, а структура S — индексное множество.

TM.UPD8 $reppar(A, S, upd(A, I, exp)) = UPD(A, S, exp)$.

Доказательство. Пусть

$$vec(S) = [s_1, \ldots, s_n],\ vec(S_j) = [s_1, \ldots, s_j]\ (j = 1, \ldots, n) \text{ и } empty(S_0)$$

для подходящих структур $S_0, S_1, \ldots, S_n$. Достаточно показать, что

$$\forall j (0 \le j \le n \to reppar(A, S_j, upd(A, I, exp)) = UPD(A, S_j, exp)).$$

Применяется индукция по $j = 0, 1, \ldots$. Базис индукции $j = 0$ очевиден. Пусть $j > 0$. Обозначим через A_j матрицу $UPD(A, S_j, exp)$. Предположим, что $reppar(A, S_{j-1}, upd(A, I, exp)) = A_{j-1}$. Тогда

$$reppar(A, S_j, upd(A, I, exp)) = upd(A_{j-1}, s_j, exp(A, s_j)).$$

Пусть $s_j = (s_j^1, s_j^2)$, тогда

$$UPD(A, S_j, exp) = UPD(A, S_{j-1} \cup \{(s_j^1, s_j^2)\}, exp),$$

причем $\{(s_j^1, s_j^2)\} = set(s_j^1, s_j^2)$. Применив теорему TM.UPD4 и аксиому AM.UPD1, получим:

$$\begin{aligned} UPD(A, S_j, exp) = &\quad UPD(UPD(A, S_{j-1}, exp), set(s_j^1, s_j^2), exp) \\ = &\quad upd(A_{j-1}, s_j, exp(s_j^1, s_j^2)). \end{aligned}$$

Рассмотрим операцию замены в массиве. Обозначим через $\{l, \ldots, r\}$ множество вида $\{l, l+1, \ldots, r\} (l \le r)$. Пусть $UPD(a, \{l, \ldots, r\}, e)$ — это массив, полученный из a в результате преобразования, где все элементы с индексом s из множества $\{l, \ldots, r\}$ заменены значением выражения $e(s)$. Аксиомы AM.UPD1′, AM.UPD2′ описывают это понятие.

AM.UPD1′
$u \in \{m, \ldots, n\} \cap \{l, \ldots, r\} \Rightarrow UPD(a[m:n], \{l, \ldots, r\}, e(s))[u] = e(u)$.

AM.UPD2′
$u \in \{m, \ldots, n\} \wedge u \notin \{l, \ldots, r\} \Rightarrow UPD(a[m:n], \{l, \ldots, r\}, e(s))[u] = a[u].$

Аксиомы AM.UPD1′ и AM.UPD2′ позволяют вычислять элементы в преобразованном массиве.

3.4. Операции перестановки

Рассмотрим вначале операцию перестановки двух строк матрицы [7]. Пусть $per(M, i, j, m, n)$ есть матрица, полученная из матрицы M в результате перестановки i-й и j-й строк, взятых от m-го до n-го столбца. Формальное определение операции per дается аксиомами AM.PER1–AM.PER3, а ее свойства выражаются теоремами TM.PER1–TM.PER6 [7].

Следующая теорема говорит о том, что если дважды поменять местами одни и те же строки матрицы, то получится исходная матрица.

TM.PER7 $(row(i, m, n) \cup row(j, m, n)) \subseteq def(M) \Rightarrow$
$per(per(M, i, j, m, n), i, j, m, n) = M.$

Рассмотрим операцию перестановки двух столбцов матрицы. С помощью аксиом AM.PERC1 – AM.PERC3 вводится понятие перестановки i-го и j-го столбцов с m-й по n-ю строку $perc(M, i, j, m, n)$.

AM.PERC1
$(col(i, m, n) \cup col(j, m, n)) \subseteq def(M) \Rightarrow def(per(M, i, j, m, n)) = def(M).$

AM.PERC2
$(col(i, m, n) \cup col(j, m, n)) \subseteq def(M) \wedge (i \neq v \wedge j \neq v \vee u < m \vee u > n) \Rightarrow$
$perc(M, i, j, m, n)[u, v] = M[u, v].$

AM.PERC3
$(col(i, m, n) \cup col(j, m, n)) \subseteq def(M) \wedge m \leq u \leq n \Rightarrow$
$perc(M, i, j, m, n)[u, i] = M[u, j] \wedge perc(M, i, j, m, n)[u, j] = M[u, i].$

Рассмотрим операцию перестановки элементов одномерного массива. Функцию перестановки элементов i и j одномерного массива a обозначим $per(a, i, j)$. Эту функцию описывают аксиомы AM.PER1′, AM.PER2′.

AM.PER1′ $def(a) = \{l, \ldots, r\} \wedge l \leq m \leq r \wedge l \leq n \leq r \Rightarrow$
$per(a, m, n)[m] = a[n] \wedge per(a, m, n)[n] = a[m].$

AM.PER2′
$def(a) = \{l, \ldots, r\} \wedge l \le m \le r \wedge l \le n \le r \wedge i \ne m \wedge i \ne n \Rightarrow$
$per(a, m, n)[i] = a[i].$

Рассмотрим предикат перестановки отрезка натурального ряда. Пусть $nper(\sigma, k, l)$ означает предикат "σ — перестановка отрезка натурального ряда от k до l включительно". Отрезок натурального ряда от l до r, представленный в виде одномерного массива, обозначим через $nat(l, r)$. Формальное определение понятия перестановки отрезка натурального ряда дают следующие аксиомы:

AM.NPER1
$nper(\sigma, l, r) \equiv (\forall u \in \{l, \ldots, r\}\ \sigma[u] \in \{l, \ldots, r\} \wedge \sigma[i] = \sigma[j] \Leftrightarrow i = j).$

AM.NPER2
$nper(\sigma, l, r) \wedge l \le m \le r \wedge l \le n \le r \Rightarrow nper(per(\sigma, m, n), l, r).$

AM.NPER3 $nper(\sigma, l, r) \wedge \forall i \in \{l, \ldots, r\}(\sigma[i] = i) \Rightarrow \sigma = nat(l, r).$

TM.NPER1
$nper(\sigma, l, r) \wedge l < k \le r \Rightarrow \bigcup_{i=l}^{k-1} set(\alpha, \sigma[i]) \cup \bigcup_{i=k}^{r} set(\alpha, \sigma[i]) = row(\alpha, l, r).$

TM.NPER2 $nper(\sigma, l, r) \wedge l < k \le r \Rightarrow$
$\bigcup_{i=l}^{k-1} col(\sigma[i], p, q) \cup \bigcup_{i=k}^{r} col(\sigma[i], p, q) = mat(p, q, l, r).$

Аксиома AM.NPER1 дает определение перестановки отрезка натурального ряда. Аксиома AM.NPER2 говорит, что если поменять местами два элемента перестановки, то истинность $nper$ сохранится, AM.NPER3 вводит понятие тождественной перестановки. Справедливость теоремы TM.NPER1 следует из аксиом AM.IND1 и AM.NPER1, а теоремы TM.NPER2 — из аксиом AM.IND3, AM.NPER1.

Рассмотрим понятие перестановки всех строк матрицы.

Через $perm(M, \sigma, \tau, m, n)$, где σ, τ являются перестановками отрезка натурального ряда от k до l, обозначим матрицу, полученную из матрицы M в результате преобразования, где на место $\tau[u]$-й строки (от m-го до n-го столбца) ставится $\sigma[u]$-я строка (от m-го до n-го столбца). Формальное определение операции $perm$ дают следующие аксиомы:

AM.PERM1
$mat(k, l, m, n) \subseteq def(M) \wedge (v < m \vee v > n) \wedge nper(\sigma, k, l) \wedge nper(\tau, k, l) \Rightarrow$
$perm(M, \sigma, \tau, m, n)[u, v] = M[u, v].$

AM.PERM2

$$mat(k,l,m,n) \subseteq def(M) \wedge (k \le u \le l \wedge m \le v \le n) \wedge nper(\sigma,k,l) \wedge nper(\tau,k,l) \Rightarrow perm(M,\sigma,\tau,m,n)[\tau[u],v] = M[\sigma[u],v].$$

Аксиомы AM.PERM1, AM.PERM2 позволяют вычислить элементы преобразованной матрицы.

TM.PERM1

$$mat(k,l,m,n) \subseteq def(M) \wedge nper(\sigma,k,l) \wedge nper(\tau,k,l) \Rightarrow def(perm(M,\sigma,\tau,m,n)) = def(M).$$

TM.PERM2 $\quad nper(\sigma,l,r) \Rightarrow perm(M,\sigma,\sigma,l,r) = M.$

TM.PERM3

$$nper(\sigma,l,r) \wedge nper(\tau,l,r) \wedge l \le i \le r \wedge l \le j \le r \wedge n \le m \Rightarrow perm(M,\sigma,per(\tau,i,j),n,m) = perm(per(M,i,j,n,m),\sigma,\tau,n,m).$$

Теорема TM.PERM1 задает область определения преобразованной матрицы, справедливость теоремы следует из аксиомы AM.PER1. Теорема TM.PERM2 следует из AM.EQ1, AM.PERM1, AM.PERM2, TM.PERM1. Теорема TM.PERM3 описывает взаимосвязь перестановок элементов массива τ и строк матрицы M. Справедливость теоремы вытекает из аксиом AM.EQ1, AM.PER1′, AM.PER3, AM.PERM2.

4. АКСИОМАТИЗАЦИЯ АЛГЕБРАИЧЕСКИХ ПОНЯТИЙ

Через $unt(M)$ обозначим предикат "M является единичной матрицей" [7]. Аксиома AM.UNT из [7] дает формальное определение этого предиката. Пусть $E[k:l,m:n]$ означает единичную матрицу с областью определения $mat(k,l,m,n)$.

Определитель непустой квадратной матрицы M обозначим через $|M|$. Для остальных матриц функция $|M|$ не определена. Формальное определение этой функции дают аксиомы AM.DET1–AM.DET6 из [7], а также следующие аксиомы:

AM.DET7

$$l-k = n-m \wedge m \le i \le n \wedge m \le j \le n \wedge i \ne j \Rightarrow |perc(M[k:l,m:n],i,j,k,l)| = -|M[k:l,m:n]|.$$

AM.DET8

$$l-k = n-m \wedge k \le j \le l \wedge \forall (u,v) \in row(j,m,n)(M[u,v] = 0) \Rightarrow |M[k:l,m:n]| = 0.$$

Аксиома AM.DET7 гласит, что при перестановке двух столбцов матрицы определитель меняет знак, AM.DET8 выражает достаточное условие равенства нулю определителя.

Обозначим через $sol(M)$ вектор-столбец $X[k:l]$, задающий решение системы уравнений вида $M[k:l, m:n]*X[k:l] = M[k:l, n+1:n+1]$ [7]. Эта функция определена для матрицы $M[k:l, m:n+1]$ при условии $l-k = n-m$ и $|M[k:l, m:n]| \neq 0$. Аксиомы AM.SOL1—AM.SOL4 дают формальное определение этого понятия, а одно из его свойств описывает теорема TM.SOL [7]. Следующая аксиома говорит о том, что при перестановке всех строк матрицы решение системы не меняется.

AM.SOL5

$$l-k = n-m \wedge nper(\sigma, k, l) \wedge nper(\tau, k, l) \Rightarrow$$
$$sol(M[k:l, m:n+1]) = sol(perm(M[k:l, m:n+1], \sigma, \tau, m, n+1)).$$

Для матрицы $M[k:l, m:n]$ при условии $n-m = 2l-2k+1$ и $|M[k:l, m+l-k]| \neq 0$ обозначим через $solv(M)$ матрицу $X[k:l,\ 1:l-k+1]$, которая задает решение систем линейных уравнений вида

$$M[k:l, m:m+l-k] * X[k:l, i:i] \approx M[k:l, m+l-k+i:m+l-k+i]$$

для каждого $i = 1, 2, \ldots, l-k+1$ [7]. Формальное определение этого понятия дается аксиомами AM.SOLV1–AM.SOLV4, а одно из ее свойств описывает теорема TM.SOLV [7]. Через $(M[k:l, m:n])^{-1}$ обозначим матрицу, обратную к данной матрице. Эта операция определена при $m-n = l-k$ и $|M[k:l, m:n]| \neq 0$.

Следующая аксиома описывает такой факт: если в матрице поменять местами два столбца, то произойдет перенумерация переменных и, следовательно, в матрице решений нужно поменять местами соответствующие строки.

AM.SOLV5

$$n-m = 2l-2k+1 \wedge m \leq i \leq m+l-k \wedge m \leq j \leq m+l-k \Rightarrow$$
$$solv(perc(M[k:l, m:n], i, j, k, l)) =$$
$$per(solv(M), k+i-m, k+j-m, 1, 1+l-k).$$

Введем аксиому, которая утверждает, что матрица решений не меняется при перестановке всех строк.

AM.SOLV6

$$n-m = 2l-2k+1 \wedge nper(\sigma, k, l) \wedge nper(\tau, k, l) \Rightarrow$$
$$solv(M[k:l, m:n]) = solv(perm(M[k:l, m:n], \sigma, \tau, m, n)).$$

5. ВЕРИФИКАЦИЯ ПРОГРАММЫ ОБРАЩЕНИЯ МАТРИЦЫ

5.1. Исходная аннотированная программа

Исходная программа (алгоритм 58б [1]) обращает квадратную матрицу a порядка n методом исключения по Гауссу и Жордану с выбором главного элемента по строке. Попутно вычисляется определитель det матрицы a. Если матрица вырожденная, то выполняется условный оператор, обеспечивающий выход из программы. В этом алгоритме предусмотрена оптимизация по памяти, так как не выделяется место для построения дополнительной матрицы, как это сделано в классических алгоритмах. По сравнению с [1] в программу внесены следующие изменения:

1. Один из циклов на Алголе представлен очевидным образом в виде финитной итерации.
2. Принято соглашение, что если число меньше ε, то оно равно нулю.

PROG1:
$\{P\}$ det:=1;
for j:=1 **to** n **do** z[j]:=j;
$\{inv\}$ **for** i:=1 **to** n **do**
begin k:=i; y:=a[i,i]; r:=i−1; p:=i+1;
for j:=p **to** n **do**
begin w:=a[i,j]; **if** abs(w)>abs(y) **then**
begin k:=j; y:=w **end end**;
det:=y× det; **if** k<>i **then** det:=−det;
if abs(y)=0 **then go to** L; y:=1/y;
for j:=1 **to** n **do begin** c[j]:=a[j,k]; a[j,k]:=a[j,i]; a[j,i]:=c[j];
a[j,i]:=−a[j,i]×y; a[i,j]:=a[i,j]×y; b[j]:=a[i,j] **end**;
j:=z[i]; z[i]:=z[k]; z[k]:=j; a[i,i]:=y;
for (k,j) **in** $mat(1,r,1,r) \cup mat(1,r,p,n) \cup mat(p,n,1,r) \cup mat(p,n,p,n)$ **do**
a[k,j]:=a[k,j]−b[j]×c[k] **end**
end;
for r:=1 **to** n **do begin** k:=z[r]; j:=r;
while k<>j **do begin for** i:=1 **to** n **do**
begin w:=a[j,i]; a[j,i]:=a[k,i]; a[k,i]:=w **end**;
i:=z[k]; z[k]:=z[j]; z[j]:=i; k:=z[j] **end end**;
L: $\{Q\}$,

где $P : n \geq 1 \wedge a[1:n,1:n] = a_0[1:n,1:n]$;
$Q : (a[1:n,1:n] = a_0^{-1} \wedge det = |a_0[1:n,1:n]|) \vee |a_0[1:n,1:n]| = 0$;
$inv : solv(con(a_0[1:n,1:n], E[1:n,1:n])) =$
$perm(solv(m), nat(1,n), z, 1, n) \wedge |a_0| =$
$det \times |m[1:n,1:n]| \wedge det \neq 0 \wedge nper(z,1,n)$,
если $m[1:n,1:2\times n] = con(E[1:n,1:i-1], a[1:n,i:n],$
$a[1:n,1:i-1], E[1:n,i:n])$.

5.2. Преобразование исходной программы

Заметим, что для аннотирования данной программы используется только один инвариант внешнего цикла. Для элиминации инвариантов пяти циклов применяются предварительные преобразования. Введем следующие обозначения. Пусть PROG1.1 имеет вид:

for j:=p **to** n **do**
begin w:=a[i,j]; **if** abs(w)>abs(y) **then begin** k:=j; y:=w **end end**;

Пусть PROG1.2 имеет вид:

for j:=1 **to** n **do begin** c[j]:=a[j,k]; a[j,k]:=a[j,i]; a[j,i]:=c[j]
a[j,i]:=−a[j,i]×y; a[i,j]:=a[i,j]×y; b[j]:=a[i,j] **end**;

Пусть PROG1.3 имеет вид:

for r:=1 **to** n **do begin** k:=z[r]; j:=r;
while k<>j **do begin**
for i:=1 **to** n **do begin** w:=a[j,i]; a[j,i]:=a[k,i]; a[k,i]:=w **end**;
i:=z[k]; z[k]:=z[j]; z[j]:=i; k:=z[j] **end end**.

Рассмотрим преобразования программы PROG1.

Программа PROG1.1 очевидным образом эквивалентна итерации PROG1.1′:

for j **in** S **do** $(w,k,y) := body(a,k,y,j)$ **end**,
где $S = [p, p+1, \ldots, n]$,
$body(a,k,y,j)$=**if** $abs(a[i,j]) > abs(y)$ **then** $(a[i,j], j, a[i,j])$ **else** $(a[i,j], k, y)$.

Программу PROG1.2 разбиваем на пять циклов, образующих эквивалентную программу PROG1.2′:

for j:=1 **to** n **do** c[j]:=a[j,k];
for j:=1 **to** n **do begin** w:=a[j,k]; a[j,k]:=a[j,i]; a[j,i]:=w **end**;
for j:=1 **to** n **do** a[j,i]:=−a[j,i]×y;
for j:=1 **to** n **do** b[j]:=a[i,j];
for j:=1 **to** n **do** a[i,j]:=a[i,j]×y;

Наша цель — заменить программу PROG1.3 эквивалентным **forpar**-оператором PROG1.3′. Для этого даются аннотации, относительно которых доказывается частичная корректность PROG1.3 и PROG1.3′.

$\{P'\}$ $\{inv_1\}$ **for** r:=1 **to** n **do begin** k:=z[r]; j:=r;
$\{inv_2\}$ **while** k<>j **do begin**
for i:=1 **to** n **do begin** w:=a[j,i]; a[j,i]:=a[k,i]; a[k,i]:=w **end**;
i:=z[k]; z[k]:=z[j]; z[j]:=i; k:=z[j] **end end** $\{Q'\}$,

где

$P' : a = a_0 \wedge z = z_0 \wedge nper(z_0, 1, n)$;
$Q' : \forall (i,k) \in mat(1:n, 1:n)(a[z_0[i], k] = a_0[i, k])$;
$inv_1 : perm(a, nat(1,n), z, 1, n) = perm(a_0, nat(1,n), z_0, 1, n) \wedge$
$\forall l \in \{1, \ldots, r-1\}(z[l] = l) \wedge nper(z, 1, n)$;
$inv_2 : inv_1 \wedge k = z[r]$.

Список условий корректности для PROG1.3:

$\beta_1 : P' \to inv_1(r \leftarrow 1)$;
$\beta_2 : inv_1 \wedge r = n+1 \to Q'$;
$\beta_3 : inv_1 \wedge 1 \le r \le n \to inv_2(k \leftarrow z[r])$;
$\beta_4 : inv_2 \wedge k \ne r \to inv_2(k \leftarrow z[r], z \leftarrow per(z, k, r), a \leftarrow per(a, k, r, 1, n))$;
$\beta_5 : inv_2 \wedge k = r \to inv_1(r \leftarrow r+1)$.

Докажем частичную корректность PROG1.3′.

$\{P'\}$ **forpar** (r,k) **in** mat(1,n,1,n) **do**
a[z[r],k]:=a[r,k] **end** $\{Q'\}$.
Условие корректности имеет вид:
$\beta : P' \to Q'(a \leftarrow reppar(a, mat(1,n,1,n), upd(a, (z[r], k), a[r, k])))$.

Справедливость β очевидным образом вытекает из следствия 2. Докажем условия корректности β_2 и β_4. Доказательство остальных условий очевидно.

Доказательство условия β_2. Один из членов посылки имеет вид

$$\forall l \in \{1, \ldots, n\}(z[l] = l),$$

значит по аксиоме AM.NPER3 имеем $z = nat(1, n)$. Тогда

$$\begin{aligned} perm(a_0, nat(1,n), z_0, 1, n) &= perm(a_0, nat(1,n), z, 1, n) \\ &= perm(a, nat(1,n), nat(1,n), 1, n) \\ &= a. \end{aligned}$$

Первое равенство встречается в посылке, последнее следует из теоремы TM.PERM2. Отсюда $perm(a_0, nat(1,n), z_0, 1, n)[z_0[i], k] = a[z_0[i], k]$.

По аксиоме AM.PERM2 получаем:

$$perm(a_0, nat(1,n), z_0, 1, n)[z_0[i], k] = a_0[i, k].$$

Таким образом, $a[z_0[i], k] = a_0[i, k]$.

Доказательство условия β_4. Справедливость $nper(per(z, k, r), 1, n)$ следует из аксиомы AM.NPER2.

Далее хотим показать, что

$$perm(per(a, k, z, 1, n), nat(1,n), per(z, k, r), 1, n) = perm(a_0, nat(1,n), z_0, 1, n).$$

Из теорем TM.PERM3 и TM.PER7 получим:

$$perm(per(a, k, z, 1, n), nat(1,n), per(z, k, r), 1, n) = perm(a, nat(1,n), z, 1, n).$$

А из посылки условия корректности имеем:

$$perm(a_0, nat(1,n), z_0, 1, n) = perm(a, nat(1,n), z, 1, n).$$

5.3. Вывод условий корректности

Обозначим через PROG1′ программу, полученную из программы PROG1 заменой циклов PROG1.i на программу PROG1.i′ для $i = 1, 2, 3$. Как показано в в 5.2, PROG1 эквивалентна PROG1′. Для вывода и преобразований условий корректности программы PROG1′ используются правила вывода rl2, R.FORT2, R.FORT1, R.FORTT1 [7], правила вывода, аналогичные R.FORT1 и R.FORT3 [7], а также теоремы 1, 3 и теорема TM.UPD8.

Список условий корректности:

α_1 : $P \to inv(i \leftarrow 1, z \leftarrow UPD(z, \{1 \dots n\}, s), det \leftarrow 1)$;

α_2 : $inv \wedge i = n + 1 \to Q(a \leftarrow reppar(a, mat(1, n, 1, n), upd(a, (z[r], k), a[r, k])))$;

α_3 : $inv \wedge 1 \le i \le n \to (k \ne i \to (abs(y) \ne 0 \to inv(i \leftarrow i + 1, a \leftarrow a_1, a \leftarrow a_2, z \leftarrow per(z, i, k), b \leftarrow b_1, a \leftarrow a_3, a \leftarrow a_4, c \leftarrow c_1, a \leftarrow a_5, y \leftarrow 1/y))(det \leftarrow -det))(det \leftarrow y \times det, (k, y) \leftarrow rep((w, i, a[i, i]), [p, \dots, n], body), p \leftarrow i + 1, r \leftarrow i - 1)$;

$$\begin{aligned}
\alpha_4: &\ inv \wedge 1 \le i \le n \to (k = i \to (abs(y) \ne 0 \to inv(i \leftarrow i+1, a \leftarrow a_1,\\
&\ a \leftarrow a_2, z \leftarrow per(z,i,k), b \leftarrow b_1, a \leftarrow a_3, a \leftarrow a_4, c \leftarrow c_1, a \leftarrow a_5,\\
&\ y \leftarrow 1/y)))(det \leftarrow y \times det, (k,y) \leftarrow rep((w,i,a[i,i]), [p,\ldots,n],\\
&\ body), p \leftarrow i+1, r \leftarrow i-1);\\
\alpha_5: &\ inv \wedge 1 \le i \le n \to (k \ne i \to (abs(y) = 0 \to Q)(det \leftarrow -det))\\
&\ (det \leftarrow y \times det, (k,y) \leftarrow rep((w,i,a[i,i]), [p,p+1,\ldots,n], body),\\
&\ p \leftarrow i+1, r \leftarrow i-1);\\
\alpha_6: &\ inv \wedge 1 \le i \le n \to (k = i \to (abs(y) = 0 \to Q))(det \leftarrow y \times det,\\
&\ (k,y) \leftarrow rep((w,i,a[i,i]), [p,\ldots,n], body), p \leftarrow i+1, r \leftarrow i-1);
\end{aligned}$$

где

$$\begin{aligned}
a_1 &= UPD(a, mat(1,r,1,r) \cup mat(1,r,p,n) \cup\\
&\ mat(p,n,1,r) \cup mat(p,n,p,n), a[s,t] - b[s] \times c[t]);\\
a_2 &= upd(a,(i,i),y) = UPD(a, set(i,i), y);\\
a_3 &= UPD(a, row(i,1,n), a[i,t] \times y);\\
a_4 &= UPD(a, col(i,1,n), -a[s,i] \times y);\\
a_5 &= perc(a,i,k,1,n);\\
b_1 &= UPD(b, \{1,\ldots,n\}, a[i,t] \times y);\\
c_1 &= UPD(c, \{1,\ldots,n\}, a[s,k]).
\end{aligned}$$

5.4. Доказательство условий корректности

Поскольку условия α_3–α_6 содержат операцию замены

$$rep((w,i,a[i,i]), S, body)$$

(сокращенно $rep(S)$), то для их доказательства будет использоваться следующее свойство:

$$\begin{aligned}
prop(rep(S)) = (y = a[i,k] \to (\forall \alpha \in con(k,S)(abs(rep_y(S)) \ge abs(a[i,\alpha]))\\
\wedge\ rep_y(S) = a[i, rep_k(S)] \wedge rep_k(S) \in con(k,S))).
\end{aligned}$$

Лемма1. Свойство $prop(rep(S)$ справедливо.

Доказательство. Для доказательства леммы воспользуемся принципом индукции 1.

При $empty(S)$ доказательство $prop(rep(S))$ очевидно. Докажем данное свойство для $\neg empty(S)$. Введем обозначение $S_x = [p, p+1, \ldots, x]$. Пусть выполнено $prop(rep(S_{x-1}))$. Имеем:

$$(rep(S_x) = body(a, rep_k(S_{x-1}), rep_y(S_{x-1}), last(S_x)).$$

Если $abs(a[i,x]) > abs(rep_y(S_{x-1}))$, то $rep_y(S_x) = a[i,x]$ и $rep_k(S_x) = x$. Но $(\forall\alpha \in con(k, S_{x-1})(abs(rep_y(S_{x-1})) \geq abs(a[i,\alpha]))$. Отсюда

$$a[i, rep_k(S_x)] = a[i,x] = rep_y(S_x)$$

и

$$(\forall\alpha \in con(k, S_x)(abs(rep_y(S_x)) \geq abs(a[i,\alpha])).$$

Случай $abs(a[i,x]) \leq rep_y(S_{x-1})$ рассматривается аналогично.

5.4.1. Условия α_1 и α_2

Условие α_1 непосредственно следует из очевидного факта

$$E[1:n, 1:0] = a[1:n, 1:0] = \Omega$$

и аксиомы AM.CON5.

Из посылки α_2 имеем:

$$solv(con(a_0[1:n, 1:n], E[1:n, 1, n])) = \\ perm(solv(con(E[1:n, 1:n], a[1:n, 1:n])), nat(1,n), z, 1, n).$$

По аксиоме AM.SOLV1 получаем:

$$solv(con(a_0[1:n, 1:n], E[1:n, 1, n])) = a_0^{-1},$$

а по теореме TM.SOLV1 заключаем, что

$$solv(con(E[1:n, 1:n], a[1:n, 1:n])) = a.$$

Мы получили $a_0^{-1} = perm(a, nat(1,n), z, 1, n)$. Но

$$reppar(a, mat(1,n,1,n), upd(a, (z[r], k), a[r,k])) = perm(a, nat(1,n), z, 1, n)$$

по определению операций *reppar* и *perm*. Таким образом, мы доказали, что $a_0^{-1} = reppar(a, mat(1,n,1,n), upd(a, (z[r], k), a[r,k]))$. Далее, так как $i = n+1$, то $m[1:n, 1:n] = E[1:n, 1:n]$, поэтому $|m[1:n, 1:n]| = 1$. Следовательно, из посылки условия корректности получаем $|a_0| = det$.

5.4.2. Условия α_3 и α_4

Докажем вначале условие α_3.

Справедливость $nper(per(z, i, rep_k(S)), 1, n)$ следует из аксиомы AM.NPER2. Введем обозначения:

$m(a,i) = con(E[1:n,1:i-1], a[1:n,i:n], a[1:n,1:i-1], E[1:n,i:n])$;
$a' = a_1(a_2(a_3(a_4(a_5))))$, где $a_i(a_j) = a_i(a \leftarrow a_j)(i,j = 1,2,3,4,5)$.

Из теоремы TM.UPD2 имеем $a' = a''$, где

$$\begin{array}{rcl} a'' & = & a_1(a_6(a_2(a_3(a_5)))), \\ a_6 & = & UPD(a, col(i,1,n) \setminus set(i,i), -c[s]/a[i,k]). \end{array}$$

Докажем, что

$$solv(con(a_0[1:n,1:n], E[1:n,1:n])) = perm(solv(m(a'', i+1)), nat(1,n), per(z,i,rep_k(S)), 1, n).$$

Из посылки условия корректности имеем:

$$solv(con(a_0[1:n,1:n], E[1:n,1:n])) = perm(solv(m(a,i)), nat(1,n), z, 1, n).$$

Благодаря TM.PERM3 достаточно доказать, что

$$solv(m(a,i)) = per(solv(m(a'', i+1)), i, rep_k(S), 1, n). \tag{4}$$

Идея доказательства (4) состоит в том, чтобы над матрицей $m(a,i)$ делать преобразования, которые приводят к матрице

$$per(solv(m(a'', i+1)), i, rep_k(S), 1, n).$$

Вначале положим $B_1 = perc(m(a,i), i, rep_k(S), 1, n)$. По аксиоме AM.SOLV5 получим:

$$solv(m(a,i)) = per(solv(B_1), i, rep_k(S), 1, n). \tag{5}$$

Но $B_1 = m(a_5, i)$.

Далее положим $B_2 = UPD(B_1, row(i,1,2n), B_1[i,t]/B_1[i,i])$. По аксиоме AM.SOLV3 имеем:

$$solv(B_2) = solv(B_1). \tag{6}$$

Заметим, что

$$\begin{array}{rcl} B_2 = & & UPD(B_1, set(i,i), 1, row(i, i+1, n+i-1), \\ & & B_1[i,t]/a[i,k], set(i, n+i), 1/a[i,k]) \\ & = & m'(a_2(a_3(a_5)), i), \end{array}$$

где $m'(a,i) = UPD(m(a,i), set(i,i), 1, set(i, i+n), a[i,i])$.

После этого положим

$$B_3 = UPD(B_2, mat(1, i-1, 1, 2n) \cup mat(i+1, n, 1, 2n), B_2[s,t] - B_2[s,i] \times B_2[i,t]).$$

Дважды применив аксиому AM.SOLV4, получим:

$$solv(B_3) = solv(B_2). \tag{7}$$

Преобразуем B_3 следующим образом:

$$\begin{aligned}
B_3 = \;& UPD(B_2, mat(1, i-1, i, n) \cup mat(i+1, n, i, n) \cup mat(1, i-1, n+1, \\
& n+i) \cup mat(i+1, n, n+1, n+i), B_2[s,t] - B_2[s,i] \times B_2[i,t])) \\
= \;& UPD(B_2, col(i, 1, i-1) \cup col(i, i+1, n), 0, mat(1, i-1, i+1, n) \cup \\
& mat(i+1, n, i+1, n), a_2(a_3(a_5))[s,t] - b[s] \times c[t], mat(1, i-1, \\
& n+1, n+i-1) \cup mat(i+1, n, n+1, n+i-1)), \\
& a_2(a_3(a_5))[s, t-n] - b[s] \times c[t-n], col(n+i, 1, i-1) \cup \\
& col(n+i, i+1, n), c[s]/a[i,k]) \\
= \;& m(a'', i+1).
\end{aligned}$$

Следовательно, (4) вытекает из (5)–(7).

Докажем, что $|a_0| = -(det \times |m(a'', i+1)[1:n, 1:n]|)/a[i,k]$. Из посылки условия корректности имеем: $|a_0| = det \times |m(a,i)[1:n, 1:n]|$. То есть нам достаточно доказать:

$$|m(a,i)[1:n, 1:n]| = -(|m(a'', i+1)[1:n, 1:n]|)/a[i,k]. \tag{8}$$

Доказательство аналогично условию (4).

Положим $C_1 = perc(m(a,i)[1:n, 1:n], i, rep_k(S), 1, n)$. По аксиоме AM.DET7 имеем:

$$|m(a,i)[1:n, 1:n]| = -|C_1|. \tag{9}$$

Заметим, что $C_1 = m(a_5, i)[1:n, 1:n]$.

Далее положим $C_2 = UPD(C_1, row(i, 1, n), C_1[i,t]/C_1[i,i])$. Из аксиомы AM.DET4 и так как $C_1[i,i] = a_1[i,i] = a[i,k]$ имеем:

$$|C_2| = |C_1|/a[i,k]. \tag{10}$$

Преобразуем C_2 следующим образом:

$$\begin{aligned}
C_2 = \;& UPD(C_1, row(i,i,n), C_1[i,t]/C_1[i,i]) \\
= \;& UPD(C_1, set(i,i), 1, row(i, i+1, n), a_1[i,t]/a_1[i,i]) \\
= \;& m'(a_3(a_5), i)[1:n, 1:n].
\end{aligned}$$

Но так как $set(i, n+i) \notin mat(i, n, i, n)$ и

$$(col(n+i, 1, i-1) \cup col(n+i, i+1, n)) \notin mat(i, n, i, n),$$

то по аксиоме AM.UPD4 имеем:

$$\begin{aligned}
m'(a_3(a_5), i)[1:n, 1:n] = \;& m'(a_2(a_3(a_5)), i)[1:n, 1:n] \\
= \;& m'(a_6(a_2(a_3(a_5))), i)[1:n, 1:n].
\end{aligned}$$

Затем положим

$$C_3 = UPD(C_2, mat(1, i-1, 1, n) \cup mat(i+1, n, 1, n), \\ C_2[s,t] - C_2[s,i] \times C_2[i,t]).$$

По аксиоме AM.DET6 имеем:

$$|C_3| = |C_2|. \tag{11}$$

Преобразуем C_3:

$$\begin{aligned} C_3 = \;& UPD(C_2, col(i,1,i-1) \cup col(1,i+1,n), 0, mat(1,i-1,i+1,n) \cup \\ & mat(i+1,n,i+1,n), a_6(a_2(a_3(a_5)))[s,t] - c[s] \times b[t]) \\ = \;& m(a_1(a_6(a_2(a_3(a_5)))), i+1). \end{aligned}$$

Следовательно, справедливость (8) вытекает из (9)–(11).

Доказательство условия α_4 аналогично доказательству α_3.

5.4.3. Условия α_5 и α_6

Покажем справедливость α_5. Хотим доказать, что

$$|a_0[1:n, 1:n]| = 0.$$

Имеем $abs(rep_y(S)) = 0$, тогда из леммы 1 следует, что $a[i,\alpha] = 0$ для всех $\alpha = i, \ldots, n$. Значит, $m(a,i)[i,\alpha] = 0$ для всех $\alpha = 1, \ldots, n$. По аксиоме AM.DET8 заключаем, что $|m(a,i)[1:n, 1:n]| = 0$. Из посылки следует, что $|a_0[1:n, 1:n]| = 0$.

Доказательство условия α_6 аналогично доказательству α_5.

6. ВЕРИФИКАЦИЯ ПРОГРАММЫ РЕШЕНИЯ СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ

6.1. Исходная аннотированная программа

Данная программа решает систему n линейных уравнений методом Гаусса без обратного хода с выбором главного элемента по всей матрице. Расширенная матрица системы имеет вид $M[1:n, 1:n+1]$. Если матрица $M[1:n, 1:n]$ вырожденная, то выполняется условный оператор, обеспечивающий выход из программы. Задача упомянута в [3], здесь предлагается одно из возможных ее решений.

PROG2:
$\{P\}$ r:=false;
for k:=1 **to** n **do begin** v[k]:=k; w[k]:=k **end**; w[n+1]=n+1;
$\{inv\}$ **for** k:=1 **to** n **do begin** m:=0;
for i:=k **to** n **do for** j:=k **to** n **do**
if abs(M[v[i],w[i]])>abs(m) **then begin** m:=M[v[i],w[j]]; p:=i; q:=j **end**;
if m=0 **then begin** r:=true; **go to** L **end**;
h:=v[k]; v[k]:=v[p]; v[p]:=h; h:=w[k]; w[k]:=w[q]; w[q]:=h;
for j:=k **to** n+1 **do** M[v[k],w[j]]:=M[v[k],w[j]]/m;
for i:=1 **to** v[k]−1 **do for** j:=n+1 **downto** k **do**
M[i,w[j]]:=M[i,w[j]]−M[v[k],w[j]]×M[i,w[k]];
for i:=v[k]+1 **to** n **do for** j:=n+1 **downto** k **do**
M[i,w[j]]:=M[i,w[j]]−M[v[k],w[j]]×M[i,w[k]] **end**;
for z:=1 **to** n **do begin** k:=w[z]; j:=v[z];
while k <> j **do begin**
h:=M[k]; M[k]:=M[z]; M[z]:=h; h:= w[k]; w[k]:=w[z]; w[z]:=h; k:=w[z] **end**
end
L: $\{Q\}$,

где

$$
\begin{aligned}
P:&\quad n \geq 1 \wedge M[1:n,1:n+1] = M_0[1:n,1:n+1];\\
Q:&\quad r \wedge |M_0[1:n,1:n]| = 0 \vee\\
&\quad \neg r \wedge sol(M_0[1:n,1:n+1]) = M[1:n,n+1,n+1];\\
inv:&\quad \neg r \wedge sol(M_0) = sol(M)\\
&\quad \wedge(\forall(s,t) \in \cup_{i=1}^{k-1}(col(w[i],1,n) \setminus set(v[i],w[i]))(M[s,t]=0)\\
&\quad \wedge\forall(s,t) \in \cup_{i=1}^{k-1} set(v[i],w[i])(M[s,t]=1) \wedge nper(v,1,n)\\
&\quad \wedge nper(w,1,n+1) \wedge w[n+1] = n+1.
\end{aligned}
$$

6.2. Преобразование исходной программы

Заметим, что для аннотирования данной программы используется только один инвариант внешнего цикла. Для элиминации инвариантов семи циклов используются предварительные преобразования. Введем следующие обозначения. Пусть PROG2.1 имеет вид:

for k:=1 **to** n **do begin** v[k]:=k; w[k]:=k **end**.

Пусть PROG2.2 имеет вид:

for i:=k **to** n **do for** j:=k **to** n **do**
if abs(M[v[i],w[i]])>abs(m) **then begin** m:=M[v[i],w[j]]; p:=i; q:=j **end**.

Пусть PROG2.3 имеет вид:

for j:=k **to** n+1 **do** M[v[k],w[j]]:=M[v[k],w[j]]/m.

Пусть PROG2.4 имеет вид:

for i:=1 **to** v[k]−1 **do for** j:=n+1 **downto** k **do**
M[i,w[j]]:=M[i,w[j]]−M[v[k],w[j]]×M[i,w[k]].

Пусть PROG2.5 имеет вид:

for i:=v[k]+1 **to** n **do for** j:=n+1 **downto** k **do**
M[i,w[j]]:=M[i,w[j]]−M[v[k],w[j]]×M[i,w[k]].

Пусть PROG2.6 имеет вид:

for z:=1 **to** n **do begin** k:=w[z]; j:=v[z];
while k <> j **do begin**
h:=M[k]; M[k]:=M[z]; M[z]:=h; h:= w[k]; w[k]:=w[z]; w[z]:=h; k:=w[z] **end**
end.

Рассмотрим преобразования программы PROG2.

Программу PROG2.1 разбиваем на два цикла, образующих эквивалентную программу PROG2.1′:

for k:=1 **to** n **do** v[k]:=k;
for k:=1 **to** n **do** w[k]:=k.

Программа PROG2.2 очевидным образом эквивалентна итерации PROG2.2′:

for (i,j) **in** S **do** (m,p,q):=$body(M,(m,p,q),(i,j))$ **end**,
где $S = mat(k,n,k,n)$,
$body(M,(m,p,q),(i,j))$=**if** $abs(M[v[i],w[i]]) > abs(m)$
then $(M[v[i],w[j]],i,j)$ **else** (m,p,q).

Преобразуем программу PROG2.3 к эквивалентной программе PROG2.3′:

for (x,y) **in** T_1 **do**
M[x,y]:=M[x,y]/m, где
$T_1 = \cup_{j=k}^{n+1} set(v[k],w[j])$.

Далее, преобразуем программу PROG2.4 к эквивалентной программе PROG2.4′:

for (x,y) **in** T_2 **do**
M[i,w[j]]:=M[i,w[j]]−M[v[k],w[j]]×M[i,w[k]], где
$T_2 = \cup_{i=1}^{v[k]-1}(\cup_{j=k}^{n+1} set(v[i],w[j]))$.

С помощью теоремы AM.IND2 упростим множество T_2:

$$\begin{aligned} T_2 = &\ \cup_{i=1}^{v[k]-1}(\cup_{j=k}^{n+1} set(v[i], w[j])) \\ = &\ \cup_{j=k}^{n+1}(\cup_{i=1}^{v[k]-1} set(v[i], w[j])) \\ = &\ \cup_{j=k}^{n+1}(col(w[j], 1, v[k]-1)) = T_2'. \end{aligned}$$

Аналогично преобразуем программу PROG2.5 к эквивалентной программе PROG2.5′:

for (x,y) **in** T_3 **do**
M[i,w[j]]:=M[i,w[j]]−M[v[k],w[j]]×M[i,w[k]], где
$T_3 = \cup_{i=v[k]+1}^{n}(\cup_{j=k}^{n+1} set(v[i], w[j]))$.

Затем с помощью теоремы AM.IND2 упростим множество T_3:

$$\begin{aligned} T_3 = &\ \cup_{i=v[k]+1}^{n}(\cup_{j=k}^{n+1} set(v[i], w[j])) \\ = &\ \cup_{j=k}^{n+1}(\cup_{i=v[k]+1}^{n} set(v[i], w[j])) \\ = &\ \cup_{j=k}^{n+1}(col(w[j], v[k]+1, n)) = T_3'. \end{aligned}$$

Наша цель — заменить программу PROG2.6 эквивалентным **forpar**-оператором PROG2.6′. Для этого даются аннотации, относительно которых доказывается частичная корректность PROG2.6 и PROG2.6′.

$\{P'\}$ $\{inv_1\}$
for z:=1 **to** n **do begin** k:=w[z]; j:=v[z];
$\{inv_2\}$
while k <> j **do begin** h:=M[k,n+1]; M[k,n+1]:=M[z,n+1]; M[z,n+1]:=h; h:= w[k]; w[k]:=w[z]; w[z]:=h; k:=w[z] **end end** $\{Q'\}$,

где

$$\begin{aligned} P' :&\ \ M = M_0 \wedge w = w_0 \wedge nper(w, 1, n) \wedge nper(v, 1, n); \\ Q' :&\ \ \forall (i, n+1) \in col(n+1, 1, n)(M[w_0[i], n+1] = M_0[v[i], n+1]); \\ inv_1 :&\ \ perm(M, v, w, n+1, n+1) = perm(M_0, v, w_0, n+1, n+1) \wedge \\ &\ \ \forall l \in \{1, \ldots, z-1\}(w[l] = v[l]) \wedge nper(w, 1, n) \wedge nper(v, 1, n); \\ inv_2 :&\ \ inv_1 \wedge k = w[z] \wedge j = v[z] \wedge 1 \le k \le n \wedge 1 \le z \le n. \end{aligned}$$

Список условий корректности для PROG2.6:

$$\begin{aligned} \beta_1 :&\ \ P' \to inv_1(z \leftarrow 1); \\ \beta_2 :&\ \ inv_1 \wedge z = n+1 \to Q'; \\ \beta_3 :&\ \ inv_1 \wedge 1 \le z \le n \to inv_2(k \leftarrow w[z], j \leftarrow v[z]); \\ \beta_4 :&\ \ inv_2 \wedge k \ne j \to inv_2(k \leftarrow w[z], w \leftarrow per(w, k, z), M \leftarrow \\ &\ \ per(M, k, z, n+1, n+1)); \\ \beta_5 :&\ \ inv_2 \wedge k = j \to inv_1(z \leftarrow z+1). \end{aligned}$$

Докажем частичную корректность PROG2.6′:

$\{P'\}$ **forpar** (i,n+1) **in** $col(n+1,1,n)$ **do**
M[w[i],n+1]:=M[v[i],n+1] **end** $\{Q'\}$,

Условие корректности имеет вид:

$$\beta : P' \to Q'(M \leftarrow reppar(M, col(n+1,1,n), \\ upd(M,(w[i],n+1),M[v[i],n+1]))).$$

Справедливость β очевидным образом вытекает из Следствия 2. Докажем условия корректности β_2 и β_4. Доказательство остальных условий очевидно.

Доказательство условия β_2. Из посылки условия корректности имеем: $\forall l \in \{1,\ldots,n\}(w[l] = v[l])$, тогда по аксиоме AM.EQ1′ $w = v$. Также из посылки следует, что

$$perm(M,v,w,n+1,n+1) = perm(M_0,v,w_0,n+1,n+1),$$

значит

$$perm(M_0,v,w_0,n+1,n+1) = perm(M,v,v,n+1,n+1) = M$$

по теореме TM.PERM2. Таким образом, мы получили:

$$perm(M_0,v,w_0,n+1,n+1)[w_0[i],n+1] = M[w_0[i],n+1].$$

Но, с другой стороны, из аксиомы AM.PERM2 следует, что

$$perm(M_0,v,w_0,n+1,n+1)[w_0[i],n+1] = M_0[v[i],n+1].$$

Значит, $M[w_0[i],n+1] = M_0[v[i],n+1]$. Во всех случаях операция перестановки строк действует на индексном множестве $col(n+1,1,n)$.

Доказательство условия β_4. Так как $nper(w,1,n)$ и $1 \le k,z \le n$, то по аксиоме AM.NPER2 $nper(per(w,k,z),1,n)$. Из определения $nper$ получим $1 \le w[k] \le n$.
Равенство

$$perm(per(M,k,z,n+1,n+1),v,per(w,k,z),n+1,n+1) = \\ perm(M,v,w,n+1,n+1)$$

следует из теорем TM.PERM4 и TM.PER7.

6.3. Вывод условий корректности

Обозначим через PROG2′ программу, полученную из программы PROG2 заменой циклов PROG2.i на программу PROG2.i′ для каждого $i = 1, 2, \ldots, 6$. Как показано в в 6.2, PROG2 эквивалентна PROG2′. Для вывода и преобразований условий корректности программы PROG2′ используются правило вывода rl2, правила вывода, аналогичные R.FORT1 и R.FORT3 [7], а также теоремы 1, 3 и теорема TM.UPD8.

Список условий корректности:

$$
\begin{array}{ll}
\alpha_1: & P \Rightarrow inv(k \leftarrow 1, r \leftarrow false, w \leftarrow upd(w, n+1, n+1), v \leftarrow v_1, \\
& w \leftarrow w_1); \\
\alpha_2: & inv \wedge 1 \le k \le n \Rightarrow inv(k \leftarrow k+1, M \leftarrow M_1, M \leftarrow M_2, M \leftarrow M_3, \\
& w \leftarrow w', v \leftarrow v')((m,p,q) \leftarrow rep(S)); \\
\alpha_3: & inv \wedge 1 \le k \le n \Rightarrow (m = 0 \Rightarrow Q(r \leftarrow true))((m,p,q) \leftarrow (rep(S)); \\
\alpha_4: & inv \wedge k = n+1 \Rightarrow Q(M \leftarrow reppar(M, col(n+1,1,n), \\
& upd(M, (w[i], n+1), M[v[i], n+1]))),
\end{array}
$$

где

$$
\begin{array}{l}
M_1 = UPD(M, \cup_{j=k}^{n+1}(col(w[j], v[k]+1, n)), M[s,t] - M[v[k],t] \times M[s, w[k]]), \\
M_2 = UPD(M, \cup_{j=k}^{n+1}(col(w[j], 1, v[k]-1)), M[s,t] - M[v[k],t] \times M[s, w[k]]), \\
M_3 = UPD(M, \cup_{j=k}^{n+1} set(v[k], w[j]), M[v[k],t]/m), \\
v_1 = UPD(v, \{1, \ldots, n\}, s), w_1 = UPD(v, \{1, \ldots, n\}, s), \\
v' = per(v, k, p), w' = per(w, k, q).
\end{array}
$$

6.4. Доказательство условий корректности

Поскольку условия α_2 и α_3 содержат операцию замены $rep((m,p,q), S, body)$ (сокращено $rep(S)$), то для их доказательства будет использоваться следующее свойство:

$$
\begin{array}{ll}
prop(rep(S)) = & \forall(\alpha, \beta) \in S(abs(rep_m(S)) \ge abs(M[v[\alpha], w[\beta]])) \wedge \\
& (rep_m(S) \ne m \rightarrow rep_m(S) = M[v[rep_p(S)], w[rep_q(S)]] \wedge \\
& (rep_p(S), rep_q(S)) \in S).
\end{array}
$$

Лемма 2. Свойство $prop(rep(S))$ справедливо.

Доказательство. Для доказательства леммы воспользуемся принципом индукции 1. При $empty(S)$ доказательство $prop(rep(S))$ очевидно. Докажем данное свойство для $\neg empty(S)$. Пусть

$$S = [(k,k) \ldots (x_0, y_0), (x, y)],$$

и пусть $prop$ выполнено для $rep(S')$, где $S' = [(k,k)\dots(x_0,y_0)]$. По определению $rep(S) = body(M, rep(S'), last(S))$, т. е., если $M[v[x], w[y]]$ по модулю меньше модуля $rep_m(S')$, то $rep(S) = rep(S')$ и $prop$ справедливо. Если же это не так, то $abs(M[v[x], w[y]]) \geq abs(M[v[\alpha], w[\beta]])$ для $(\alpha,\beta) \in S$, тогда $rep_m(S) = M[v[x], w[y]]$ и $(rep_p(S), rep_q(S)) = (x,y) \in S$.

Доказательство условия α_1. Справедливость формул $nper(v,1,n)$ и $nper(w,1,n+1)$ следует из аксиомы AM.NPER1. Справедливость кванторных формул следует из очевидного свойства

$$\cup_{i=1}^{0}(col(w[i],1,n) \setminus set(v[i],w[i])) = \cup_{i=1}^{0} set(v[i],w[i]) = \varepsilon.$$

Доказательство условия α_2. Пусть

$$M_i(M_j) = M_i(M \leftarrow M_j)(i,j = 1,2,3).$$

Докажем, что

$$\forall (s,t) \in \cup_{i=1}^{k} set(v'[i], w'[i])(M_1(M_2(M_3))[s,t] = 1).$$

Из посылки имеем:

$$\forall (s,t) \in \cup_{i=1}^{k-1} set(v[i], w[i]) M[s,t] = 1.$$

Так как

$$\cup_{i=1}^{k-1} set(v'[i], w'[i]) = \cup_{i=1}^{k-1} set(v[i], w[i])$$

и

$$\begin{aligned}(\cup_{i=1}^{k-1} set(v'[i], w'[i])) \cap (\cup_{j=k}^{n+1} set(v'[k], w'[j])) = \\ (\cup_{i=1}^{k-1} set(v'[i], w'[i])) \cap (\cup_{j=k}^{n+1} col(w[j], 1, v[k]-1)) = \\ (\cup_{i=1}^{k-1} set(v'[i], w'[i])) \cap (\cup_{j=k}^{n+1} col(w[j], v[k]+1, n)) = \quad \varepsilon,\end{aligned}$$

то по аксиоме AM.UPD4 заключаем, что

$$\forall (s,t) \in \cup_{i=1}^{k-1} set(v'[i], w'[i])(M_1(M_2(M_3))[s,t] = 1).$$

Осталось вычислить $M_1(M_2(M_3))[v'[k], w'[k]]$. Используя аксиому AM.UPD4 и $prop(rep(S))$, получим

$$\begin{aligned}M_1(M_2(M_3))[v'[k], w'[k]] = \\ M[v'[k], w'[k]]/rep_m(S) = \\ M[v[rep_p(S), w[rep_q(S)]]/M[v[rep_p(S), w[rep_q(S)]] = \quad 1.\end{aligned}$$

Член заключения

$$\forall(s,t) \in \cup_{i=1}^{k}(col(w'[i],1,n) \setminus set(v'[i],w'[i])(M_1(M_2(M_3))[s,t] = 0)$$

доказывается аналогично.

Далее докажем, что $sol(M_0) = sol(M_1(M_2(M_3)))$. Для этого достаточно доказать равенство $sol(M) = sol(M_1(M_2(M_3)))$. Из посылки имеем:

$$\forall(s,t) \in \cup_{j=1}^{k-1} set(v[k],w[k])(M[s,t] = 0).$$

Тогда

$$\forall(s,t) \in \cup_{j=1}^{k-1} set(v'[k],w'[k])(M[s,t] = 0)$$

и так как по теореме TM.NPER1 имеет место равенство

$$\bigcup_{j=1}^{k-1} \cup \bigcup_{j=k}^{n+1} set(v'[k],w'[k]) = row(v'[k],1,n+1),$$

то

$$M_3 = UPD(M, row(v'[k],1,n+1), M[v'[k],t]/rep_m(S)).$$

По аксиоме AM.SOL3 заключаем, что $sol(M_3) = sol(M)$.

Аналогично, так как

$$\forall(s,t) \in \cup_{j=1}^{k-1} set(v[k],w[k])(M[s,t] = 0),$$

то по теореме TM.NPER2 имеем:

$$M_2(M_3) = UPD(M_3, mat(1, v[k]-1, 1, n+1), \\ M_3[s,t] - M_3[v[k],t] \times M_3[s,w[k]])$$

и

$$M_1(M_2(M_3)) = UPD(M_2(M_3), mat(v[k]+1, n, 1, n+1), \\ M_2(M_3)[s,t] - M_2(M_3)[v[k],t] \times M_2(M_3)[s,w[k]]).$$

По аксиоме AM.SOL4 заключаем, что $sol(M_3) = sol(M_1(M_2(M_3)))$.

Доказательство условия α_3**.** Из посылки получаем, что

$$\forall(s,t) \in \cup_{i=1}^{k-1} set(v[k],w[i])(M[s,t] = 0),$$

а из леммы 2 и равенства нулю $rep_m(S)$ имеем:

$$\forall(s,t) \in \cup_{i=k}^{n} set(v[k],w[i])(M[s,t] = 0).$$

Таким образом, из теоремы TM.NPER1 получаем:

$$\forall(v[k],t) \in row(v[k],1,n)(M[v[k],t] = 0).$$

По аксиоме AM.DET8 получаем $|M[1:n,1:n]| = 0$, поэтому $sol(M)$ не определена. Так как $sol(M) = sol(M_0)$, то $sol(M_0)$ не определена. Значит, по определению функции sol получим $|M_0[1:n,1:n]| = 0$.

Доказательство условия α_4. Положим

$$A = reppar(M, col(n+1,1,n), upd(M,(w[i],n+1), \\ M[v[i],n+1]))[1:n,n+1,n+1].$$

Хотим показать, что $A = sol(M_0)$. Имеем:

$$\begin{aligned} sol(M_0) = \quad & sol(M) = sol(perm(M[1:n,1:n+1],v,w,1,n+1)) = \\ & sol(con(perm(M[1:n,1:n],v,w,1,n), \\ & perm(M[1:n,n+1:n+1],v,w,n+1,n+1))) = \\ & sol(con(E,A)), \end{aligned}$$

тогда по TM.SOL получим $A = sol(M_0)$.

Первое равенство следует из посылки, второе — из аксиомы AM.SOL5, третье — из аксиомы AM.CON4.

Докажем, что имеет место четвертое равенство.

1)Введем обозначение: $B = perm(M[1:n,1:n],v,w,1,n)$. Надо доказать,что $B[1:n,1:n] = E[1:n,1:n]$.

Покажем, что $B[s,t] = 0$ для $1 \le s,t \le n$, причем $s \ne t$. Из посылки имеем $nper(w,1,n)$, следовательно, можем считать, что $s = w[k]$ и $t = w[l]$ для некоторых k и l, причем $k \ne l$. С помощью аксиомы AM.PERM2 находим $B[s,t] = M[v[k],w[l]]$. А из посылки имеем $M[v[k],w[l]] = 0$.

Покажем, что $B[i,i] = 1$ для $1 \le i \le n$. Пусть $i = w[k]$, тогда $B[i,i] = B[w[k],w[k]]$ и по аксиоме AM.PERM2 получим $B[i,i] = M[v[k],w[k]]$. Из посылки следует, что $M[v[k],w[k]] = 1$.

2) Надо проверить равенство $A = perm(M,v,w,n+1,n+1)$.

По определению функции upd мы имеем: $A[w[j],n{+}1] = M[v[j],n{+}1]$, где $1 \le j \le n$, а из аксиомы AM.PERM2 следует

$$perm(M,v,w,n+1,n+1)[w[j],n+1] = M[v[j],n+1],$$

где $1 \le j \le n$, откуда и вытекает требуемое равенство.

7. ЗАКЛЮЧЕНИЕ

Таким образом, в работе описана методология верификации программ линейной алгебры, которая применяется к оптимальным программам линейной алгебры. Основная трудность верификации таких программ – синтез инвариантов циклов, которые могут быть настолько сложными, что требуется разработка специальных понятий для представления инвариантов, ориентированных на конкретные алгоритмы линейной алгебры.

Представленная методология базируется на символическом методе верификации финитных итераций, который позволяет проводить верификацию программ без синтеза инвариантов циклов [10, 11]. Важную роль в этой методологии играет операция параллельной замены, которая позволяет устранить индукцию при доказательстве условий корректности, содержащих операцию замены. Отметим, что операция параллельной замены была введена в [4] для циклов for по неупорядоченным множествам и обобщена в [6] для циклов for по линейно упорядоченным множествам. Для моделирования синхронных параллельных вычислений в [12] используется оператор, эквивалентный циклу по неупорядоченному множеству. Такой оператор цикла выражается в виде ограниченного квантора общности по переменным, удовлетворяющим булеву выражению. В [12] определяется также аналог операции параллельной замены как обобщение понятия upd.

Для простых программ линейной алгебры характерны итерации, удовлетворяющие условию Теоремы 3, что позволяет использовать вместо операции замены операцию параллельной замены и применить Следствие 2 для ее вычисления без индукции. Однако, в оптимальных программах линейной алгебры встречается немало итераций, не удовлетворяющих условию Теоремы 3. Для преодоления этой трудности в настоящей работе предложен новый оператор параллельного преобразования массивов, аксиоматическая семантика которого описана с помощью операции параллельной замены. Хотя этот оператор не встречается в исходных программах линейной алгебры, в ряде случаев удается доказать его эквивалентность данной итерации, что позволяет использовать Следствие 2 при доказательстве условий корректности. В других случаях вместо инварианта используется подходящее свойство операции замены, которое, как правило, проще инварианта. Для доказательства таких свойств успешно применяется принцип индукции 1.

Описанная методология верификации программ линейной алгебры

включает также базу знаний, состоящую из операций над массивами и понятий линейной алгебры, которая расширяет представленную в [4, 7, 8] базу знаний о программах линейной алгебры. Отметим, что описанный в разд. 5 процесс верификации программы 58б [1] проходит без существенных изменений и для программы 120б [2]. Предполагается разработать новую систему верификации программ линейной алгебры на базе описанной методологии, которая существенно расширяет систему СПЕКТР [8].

СПИСОК ЛИТЕРАТУРЫ

1. **Агеев М.И., Алик В.П. Марков Ю.И.** Библиотека алгоритмов 51б–100б. — М.: Советское радио, 1976.
2. **Агеев М.И., Алик В.П., Марков Ю.И.** Библиотека алгоритмов 101б–150б. — М.: Советское радио, 1978.
3. **Балуев А.Н. и др.** Сборник упражнений по АЛГОЛ-60. — Ленинград: Ленинградский университет, 1967.
4. **Непомнящий В.А.** Доказательство правильности программ линейной алгебры // Программирование. — 1982.— N 4. — С. 63–72.
5. **Непомнящий В.А.** Элиминация инвариантов циклов при верификации программ // Программирование — 1985. — N 3. — С. 3–13.
6. **Непомнящий В.А.** О проблемно-ориентированной верификации программ // Программирование. — 1986. — N 1. — С. 3–12.
7. **Непомнящий В.А., Рякин О.М.** Прикладные методы верификации программ. — М.: Радио и связь, 1988.
8. **Непомнящий В.А., Сулимов А.А.** Верификация программ линейной алгебры в системе СПЕКТР // Кибернетика. — 1992. — N 5. — С. 136–144.
9. **Непомнящий В.А.** Верификация программ над массивами // Системная информатика. — Новосибирск: Наука, 1993. — Т. 3. — С. 68-98.
10. **Непомнящий В.А.** Верификация финитной итерации над структурами данных // Кибернетика и системный анализ. — 1999. — № 3. — С. 25–37.
11. **Nepomniaschy V.A.** Verification of definite iteration over hierarchical data structures // Lect. Notes Comput. Sci. — 1999. — Vol. 1577. — P. 176–187.
12. **Chandy K.M., Misra J.** Parallel program design // Reading a.o.: Addison–Wesley. — 1988. — 516 p.
13. **Hoare C.A.R.** An axiomatic basis of computer programming // Comm. ACM. — 1969. — N 10. — P. 576–580.
14. **Reynolds J.C.** Reasoning about arrays // Comm. ACM. — 1979. — Vol. 22, N 5. — P. 290–299.
15. **Stark J., Ireland A.** Invariant discovery via failed proof attempts // Lect. Notes Comput. Sci. — 1999. — Vol. 1559. — P. 271–288.
16. **Stavely A.M.** Verifying definite iteration over data structures // IEEE Trans. Software Engineering. — 1995. — Vol. 21, N 6. — P. 506–514.

В. А. Непомнящий, Е. А. Мацко

ПРИМЕНЕНИЕ СИМВОЛИЧЕСКОГО МЕТОДА ЭЛИМИНАЦИИ ИНВАРИАНТОВ ЦИКЛОВ К ВЕРИФИКАЦИИ ПРОГРАММ ЛИНЕЙНОЙ АЛГЕБРЫ

Препринт
102

Рукопись поступила в редакцию 04.03.2003
Рецензент Ф. А. Мурзин
Редактор З. В. Скок

Подписано в печать 22.05.2003
Формат бумаги 60×84 1/16 Объем 2,2 уч.-изд.л., 2,4 п.л.
Тираж 50 экз.

НФ ООО ИПО "Эмари" РИЦ, 630090, г. Новосибирск, пр. Акад. Лаврентьева, 6