ПРОГРАММНО-ТЕХНИЧЕСКИЕ КОМПЛЕКСЫ

М.С. ЛЬВОВ, В.С. ПЕСЧАНЕНКО

УДК 004.421.6

МЕТОД ТРАПЕЦОИДОВ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ НЕРАВЕНСТВ И ЕГО РЕАЛИЗАЦИЯ ИНСЕРЦИОННЫМ МОДЕЛИРОВАНИЕМ

Ключевые слова: *системы линейных неравенств, алгебраическое программирование, инсерционное моделирование, система моделирования IMS.*

В настоящей статье изложен новый алгоритм решения системы линейных неравенств (СЛН), основанный на так называемом методе трапецоидов [1]. Методология алгебраического программирования, использованная в [1], заключается в конструктивном определении многосортной алгебраической системы [2], объекты которой представлены в виде выражений, а алгоритмы вычисляют значения этих выражений, т.е. строят их канонические формы [3–8]. Основная каноническая форма СЛН представляет многогранник решений СЛН в виде объединения конечного числа трапецоидов. Эта каноническая форма использует идею проектирования (элиминации квантора) методов Фурье–Моцкина [9,10], Черникова [11]. В настоящей работе определяется вторая каноническая форма СЛН, приводится алгоритм преобразования основной канонической формы СЛН во вторую каноническую форму. Эта каноническая форма представления СЛН используется как аргумент алгоритма построения множества базисных векторов многогранника решений СЛН. В заключение приводится инсерционная модель данного алгоритма.

ПРЕДСТАВЛЕНИЕ СИСТЕМ ЛИНЕЙНЫХ НЕРАВЕНСТВ В МНОГОСОРТНОЙ АЛГЕБРЕ

Линейное неравенство (ЛН) L имеет вид

$$L = a_1x_1 + a_2x_2 + \ldots + a_nx_n \le b,\ x_j \in Variable,\ a_j, b \in Coef,$$

$$A = (a_1, \ldots, a_n),\ X = (x_1, \ldots, x_n),\ L = AX \le b.$$

При этом алгебра линейных неравенств использует линейное пространство *LinComb* над упорядоченным полем *Coef*, а также алгебру переменных *Variable*. СЛН задаются логическими формулами вида $S = L_1 \& L_2 \ldots \& L_m$, где L_j — линейные неравенства.

Определение 1. Пусть x — переменная левой части ЛН с ненулевым коэффициентом и $Y = X - \{x\}$. Тогда ЛН можно преобразовать к виду

$$x \le BY + b \text{ или } x \ge BY + b. \tag{1}$$

Такую форму ЛН будем называть разрешенной относительно x (x-разрешенной). Если каждое ЛН системы, зависящее от x, представлено в разрешенной форме, то СЛН будем называть представленной в x-разрешенной форме.

Любую СЛН можно представить в x-разрешенной форме (1). Более точно СЛН можно представить в виде $LL(x) \& GL(x) \& FL$, где каждое неравенство $LL(x)$ имеет вид $x \le BY + b$, каждое неравенство $GL(x)$ имеет вид $x \ge BY + b$, каждое неравенство FL не зависит от x.

Многоосновная алгебра системы линейных неравенств состоит из следующих сортов.

LinUnEqu. Элементами этого сорта являются линейные неравенства, представленные в x-разрешеной форме

$$x_n \le a_{n-1}x_{n-1} + \ldots + a_1x_1 + b_1 \text{ или } x_n \ge a_{n-1}x_{n-1} + \ldots + a_1x_1 + b_1,$$

где x_n — старшая переменная линейного неравенства.

Многосортная операция $XCan(L)$ приводит произвольное выражение $L = B_1X + b_1 \le B_2X + b_2$ в сигнатуре алгебры линейных неравенств к канонической форме (1).

VarSegment — сорт элементарных многогранников пространства решений СЛН. Элемент этого сорта описывается конструктором $s = [L(Y), x, G(Y)]$ с семантикой $L(Y) \le x \le G(Y)$, где x — переменная, $Y = X - \{x\}$, $L(Y), G(Y)$ — линейные комбинации переменных из множества $Y \subset Variable$ с коэффициентами из $Coef$, т.е. элементами сорта $LinComb(Y, Coef)$. Элементы сорта *VarSegment* назовем x-сегментами или сегментами. Неравенство, представленное в x-разрешенной форме $x \le G(Y)$ или $x \ge L(Y)$, можно преобразовать в x-сегмент $s = [-\infty, x, G(Y)]$ или $s = [L(Y), x, +\infty]$.

Основные операции сорта — операция разбиения сегмента и обратная ей операция приведения сегмента. Они определяются соотношением $[L, x, G] = [L, x, H] + [H, x, G]$.

Trapezoid — сорт элементарных трапецоидов в пространстве решений СЛН. Для n-мерного пространства $W(X)$, где $X = \{x_1, \ldots, x_n\}$, элемент этого сорта задан конструкцией

$$T = [L_n(X_{n-1}), x_n, G_n(X_{n-1})]. \ldots .[L_2(x_1), x_2, G_2(x_1)].[L_1, x_1, G_1],$$

где $X_{n-1} = \{x_1, \ldots, x_{k-1}\}$, с семантикой

$$T = [L_n(X_{n-1}) \le x_n \le G_n(X_{n-1})] \& \ldots \& [L_2(x_1) \le x_2 \le G_2(x_1)] \& [L_1 \le x_1 \le G_1],$$

Точкой отмечен конструктор сорта *Trapezoid*.

Замечание 1. Трапецоид — выпуклая фигура в n-мерном пространстве, последовательные проекции которой на подпространства меньшей размерности — также трапецоиды. Уравнения $x_n = L(X_{n-1})$, $x_n = G(X_{n-1})$ задают нижнюю и верхнюю «шапки» трапецоида, если числовую ось Ox_n рассматривать как вертикальную. Таким образом, если трапецоид T задан формулой $T = s.T_1$, то x_n — сегмент S определяет T по координате x_n, а T_1 — проекция T на подпространство $W(X_{n-1})$. ♦

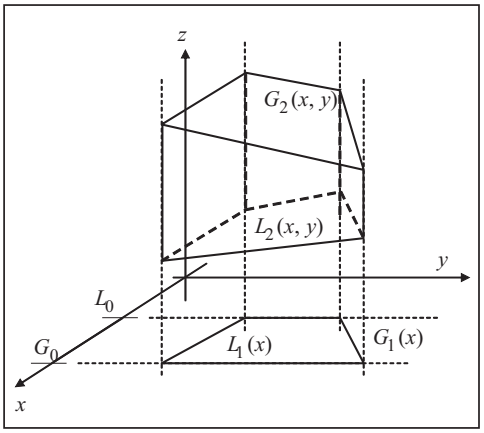

Рис. 1. Трапецоид и его проекции в трехмерном пространстве $W(x, y, z)$

Точка (отметка конструктора сорта *Trapezoid*) интерпретируется как операция пересечения трапецоидов специального вида. Введем в рассмотрение бесконечный сегмент $I(x) = [-\infty, x, +\infty]$ (рис. 1). Тогда $T = s(x_k).T_1(X_{k-1})$ можно представить в виде

$$T' = s(x_k).I(x_{k-1}). \ldots .I(x_1),\ T'' = I(x_k).T_1(X_{k-1}),\ T = T' \& T''.$$

Операции разбиения и приведения сегментов расширяются до (частичных) операций над трапецоидами

$$[L, x, G].T = [L, x, H].T + [H, x, G].T.$$

ConPol (Convex Polygon) — сорт выпуклых многогранников в пространстве решений СЛН. В n-мерном пространстве элемент этого сорта (полигон, выпуклый многогранник) задан конструкцией $P = T_1 ++ T_2 ++ \ldots ++ T_k$.

Замечание 2. Отметка ++ конструктора сорта фиксирует разбиение выпуклого многогранника на непересекающиеся либо смежные по граням трапецоиды, меньших, чем n размерностей. Линейный порядок, зафиксированный на множестве переменных *Variable*, определяет последовательность проектирований выпуклого многогранника P на подпространства все меньшей и меньшей размерности. Операция разбиения/приведения трапецоидов расширяется до операции разбиения/приведения выпуклых многогранников. Основная операция на сорте *ConPol* — пересечение двух выпуклых многогранников.

ОПЕРАЦИЯ ПЕРЕСЕЧЕНИЯ НА СОРТЕ *ConPol*

Эффективность алгоритма в целом зависит от эффективности реализации операции пересечения на сорте *ConPol*. Символ операции пересечения — знак конъюнкции &. Эта операция задана на сорте *ConPol* через ее определения на базовых сортах *VarSegment* и *Trapezoid*. Пересечение двух элементов сорта *VarSegment* определено формулой

$$[L_1(Y), x, G_1(Y)] \& [L_2(Y), x, G_2(Y)] = [\max(L_1(Y), L_2(Y)), x \min(G_1(Y), G_2(Y))].$$

При этом результат операции должен удовлетворять соотношению

$$\max(L_1(Y), L_2(Y)) \le \min(G_1(Y), G_2(Y)).$$

Однако $\max(L_1, L_2)$ и $\min(G_1, G_2)$ не принадлежат базовому сорту *LinComb*. Поэтому операция пересечения должна быть определена более точно. Действительно, операция пересечения двух трапецоидов сводится к операции пересечения трапецоида и полупространства, заданного неравенством $x \le G(Y)$ или неравенством $x \ge L(Y)$. Рис. 2 иллюстрирует различные варианты взаимного расположения границ сегментов для случая $n = 2$, при котором рассматриваемые многогранники — суть трапеции. Эти же варианты имеют место в общем случае. Соотношение $L \le_T G$ означает, что в трапецоиде $[L(X_{k-1}), x_k, G(X_{k-1})].T$ верхняя «шапка» расположена выше нижней: если верхняя и нижняя «шапки» заданы уравнениями $x_k = G(X_{k-1})$, $x_k = L(X_{k-1})$, то для любого вектора $X_{k-1}^{(0)} = (x_1^{(0)}, \ldots, x_{k-1}^{(0)}) \in T$ имеет место неравенство $L(X_{k-1}^{(0)}) \le G(X_{k-1}^{(0)})$. В канонической форме представления трапецоида эти соотношения должны выполняться для любого значения $k = 1, 2, \ldots, n$.

Рассмотрим все варианты взаимного расположения верхней и нижней шапок трапецоида $x_k = G_1(X_{k-1})$, $x_k = L_1(X_{k-1})$ при условии $L_1 \le_T G_1$ и пересекающей гиперплоскости $x_k = G(X_{k-1})$.

Вариант 1:

$$G_1 \le_T G \to s_1.T \,\&\, s = s_1.T.$$

Вариант 2:

$$(G \,\&\, G_1 \ne_T \varnothing) \,\&\, (L_1 \le_T G) \to s_1.T \,\&\, s = [L_1, x_k, G_1].T_1 ++ [L_1, x_k, G].T_2,$$

где $T_1 = \{X_{k-1} : G_1 \le_T G\}$, $T_2 = T - T_1$.

Вариант 3:

$$(G \,\&\, G_1 \ne_T \varnothing) \,\&\, (G \,\&\, L_1 \ne_T \varnothing) \to s_1.T \,\&\, s = [L_1, x_k, G_1].T_1 ++ [L_1, x_k, G].T_2,$$

где $T_1 = \{X_{k-1} : G_1 \le_T G\}$, $T_2 = \{X_{k-1} : G \le_T G_1\}$.

Вариант 4:

$$L_1 \le_T G \le_T G_1 \to s_1.T \,\&\, s = [L_1, x_k, G].T.$$

Вариант 5:

$$(G \le_T G_1) \,\&\, (L_1 \,\&\, G \ne_T \varnothing) \to s_1.T \,\&\, s = [L_1, x_k, G].T_1, \text{ где } T_1 = \{X_{k-1} : L_1 \le_T G\}.$$

Вариант 6:

$$G \le_T L_1 \to s_1.T \,\&\, s = \varnothing.$$

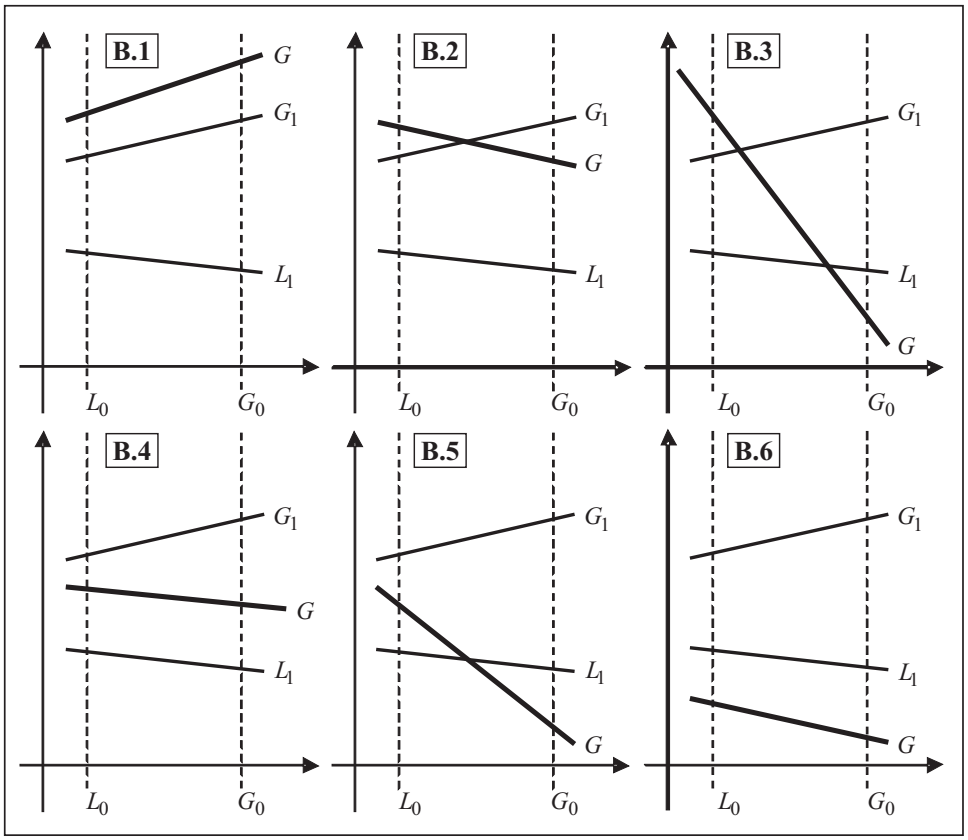

Рис. 2. Варианты пересечения трапецоида полуплоскостью

Основной источник неэффективности алгоритма, описываемого соотношениями этих вариантов, — наличие двух слагаемых в правых частях соотношений и необходимость вычислений новых трапецоидов-проекций T_1 и T_2. В варианте 6 количество трапецоидов в пересечении уменьшается, в вариантах 1 и 4 количество трапецоидов не изменяется, причем проекции остаются неизменными, в варианте 5 количество трапецоидов не изменяется, но проекция изменяется. Худшие случаи — варианты 2, 3, в которых и количество трапецоидов увеличивается, и проекции изменяются.

Алгоритм распознавания варианта. Все соотношения, определенные в вариантах, являются условными. Условия определены в терминах отношения частичного порядка $\le_T$. Именно на трапецоиде $T^{(k)}$ выполняется неравенство $G(X_{k-1}) - L(X_{k-1}) \ge 0$. В свою очередь, это означает, что

$$y_{\max} = \max_{X \in T} (L(X) - G(X)) \le 0.$$

Исходя из того, что функция $y = L(X) - G(X)$ линейна и трапецоид T — многогранная область, следует, что максимум достигается в одной из вершин трапецоида T. Наконец, форма представления трапецоида в виде последовательности его проекций на подпространства все меньшей размерности позволяет вычислить искомый максимум достаточно эффективно — за время $O(n^2)$. Покажем это.

Приведем трапецоид

$$T^{(n)} = s_n . s_{n-1} . \ldots . s_1, \; s_j = [l_j(X_{j-1}), x_j, g_j(X_{j-1})], \; j = n, n-1, \ldots, 1,$$

к виду

$$T' = s'_n . s'_{n-1} . \ldots . s'_1, \; s'_j = [0, x'_j, g'_j(X'_{j-1})]$$

преобразованиями: для всех $j = 1, 2, \ldots, n$ таких, что $l_j(X_{j-1}) \neq -\infty$, положим $h_j(X_{j-1}) = g_j(X_{j-1}) - l_j(X_{j-1})$ с последующими заменами переменных $x_j = x'_j + l_j(X_{j-1})$ в $h_j(X_{j-1})$: $g'_j(X'_j) = \mathrm{Sub}_{x_j}^{x'_j + l'_j(X'_{j-1})}(h(X_{j-1}))$. Эти преобразования выполняются на трапецоиде $T^{(n)}$ за время $O(n^2)$.

Рассмотрим следующую задачу линейного программирования (ЗЛП). Найти максимум MF линейной функции $F = c_1x_1 + \ldots + c_nx_n$ на трапецоиде $T^{(n)}$ (штрихи в обозначениях переменных в дальнейшем доказательстве опускаем). Поскольку решение ЗЛП достигается в вершинах трапецоида $T^{(n)}$, могут иметь место три случая.

1. Имеем $l_n = -\infty$, $x_n = -\infty$. Тогда если $c_n > 0$, то максимум достигается при $x_n = g_n(X_{n-1})$. Если $c_n < 0$, то $MF = +\infty$.

2. Имеем $l_n = 0$, $x_n = 0$. Тогда ЗЛП сводится к задаче размерности $n-1$: на трапецоиде $T^{(n-1)} = s_{n-1} \ldots . s_1$ найти максимум MF_1 функции $F_1 = c_1x_1 + \ldots + c_{n-1}x_{n-1}$.

3. Имеем $x_n = g_n(X_{n-1})$. Тогда ЗЛП сводится к следующей ЗЛП размерности $n-1$: на трапецоиде $T^{(n-1)} = s_{n-1} \ldots . s_1$ найти максимум MF_2 функции

$$F_2 = c_1x_1 + \ldots + c_{n-1}x_{n-1} + c_ng_n(X_{n-1}).$$

Таким образом, $MF = \max(MF_1, MF_2)$.

Поскольку на $T^{(n-1)}$ $g = g_n(X_{n-1}) \geq 0$, имеет место один из случаев:

а) $c_n < 0$, $MF = \max(MF_1, F_1 - g) = MF_1$;

б) $c_n > 0$, $MF = \max(MF_1, F_2) = MF_2$.

Итак, ЗЛП размерности n на трапецоиде T сводится к задаче размерности $n-1$ на трапецоиде $T^{(n-1)}$ за время $O(n)$. Следовательно, ЗЛП размерности n решается на трапецоиде за время $O(n^2)$.

Положим $F_1(X_{n-1}) = G(X_{n-1}) - G_1(X_{n-1})$, $F_2(X_{n-1}) = G(X_{n-1}) - L_1(X_{n-1})$. Тогда:

если $\min\limits_{X_{n-1} \in T^{(n-1)}} F_1(X_{n-1}) \geq 0$, то имеет место вариант 1;

если $\max\limits_{X_{n-1} \in T^{(n-1)}} F_1(X_{n-1}) \geq 0$, $\min\limits_{X_{n-1} \in T^{(n-1)}} F_1(X_{n-1}) \leq 0$,

$\min\limits_{X_{n-1} \in T^{(n-1)}} F_2(X_{n-1}) \geq 0$, то имеет место вариант 2.

Остальные варианты распознаются аналогично. Итак, алгоритм распознавания варианта сводится к алгоритму решения ЗЛП на трапецоиде.

КАНОНИЧЕСКИЕ ПРЕДСТАВЛЕНИЯ СИСТЕМ ЛИНЕЙНЫХ НЕРАВЕНСТВ

Как отмечалось выше, выпуклый многогранник — элемент сорта *ConPol*, представим в виде разбиения его на трапецоиды:

$$P = T_1 + + T_2 + + \ldots + + T_k. \qquad (2)$$

На множестве трапецоидов естественным образом может быть задано отношение линейного порядка, индуцированное отношениями порядка на сортах *Coef*, *Variable* и расширенное вначале на сорт *Trapezoid*, а затем и на сорт *ConPol*. Таким образом, можно считать, что $T_1 > T_2 > \ldots > T_k$, $T_j = s_{jn} . s_{jn-1} \ldots . s_{j1}$, $j = 1, 2, \ldots, k$.

Применяя соотношения

$$s.0 = 0, \; s.I = s, \; T + + 0 = 0 + + T = T,$$

можно исключить «лишние» слагаемые и сомножители. Тогда элемент P представим в канонической форме

$$P = \sum_{j=1}^{k} \prod_{i=1}^{n} S_{ji}. \tag{3}$$

Эта форма называется полиномиальной (основной) нормальной формой (ПНФ) системы линейных неравенств.

Поскольку в конструктивной алгебре сорта *ConPol* имеет место закон дистрибутивности, наряду с представлением многочлена P в виде суммы мономов — трапецоидов T_j можно определить еще одну каноническую форму — так называемую рекурсивную нормальную форму (РНФ) [1].

Вычисления полиномиальной нормальной формы. Наиболее простой подход к управлению вычислениями заключается в последовательном добавлении к уже построенной ПНФ (или РНФ) нового неравенства, представленного в x-разрешенной форме:

$$l\& P = l\& (s_1.T_1 + + \ldots + + sk.T_k) = (l\& s_1).T_1 + + \ldots + + (l\& s_k).T_k.$$

Как было показано, алгоритм перевычисляет неравенства, определяющие трапецоид, за время $O(n^2)$. Пусть $C(m, n)$ — количество трапецоидов (мономов) в текущем представлении (3). Тогда на перевычисление полинома (2) тратится $O(n^2 C(m, n))$ шагов. Поскольку алгоритм добавления неравенства повторяется m раз, то $T(m, n) = O(mn^2 C(m, n))$.

Нетрудно показать, что в разбиении многогранной области P на трапецоиды каждый трапецоид можно ассоциировать по крайней мере с одной (своей) вершиной P. Поэтому количество трапецоидов в P не превосходит количества V его вершин (0-граней). При этом необходимо учитывать и бесконечно удаленные вершины. Поэтому $T(m, n) = O(mn^2 V(m, n))$.

Поскольку трапецоид содержит 2^n вершин, эта оценка является экспоненциальной. При этом она полиномиальна относительно количества вершин многогранника.

АЛГОРИТМ ВЫЧИСЛЕНИЯ МНОЖЕСТВА БАЗИСНЫХ ВЕКТОРОВ

Пусть полигон в n-мерном пространстве задан в полиномиальной канонической форме (2), т.е. в виде разбиения на трапецоиды:

$$P = T_1 + + T_2 + + \ldots + + T_k,\ T_i = S_{i1}.S_{i2}.\ldots.S_{in},\ S_{ij} = [L_{ij}, x_j, G_{ij}].$$

Известно, что множество базисных векторов определяется множеством вершин (0-граней) полигона P. Легко увидеть, что все искомые вершины — суть вершины трапецоидов. Обратное, однако, неверно. Среди вершин трапецоидов много лишних, поскольку разбиение на трапецоиды учитывает и нижние, и верхние грани одновременно. Из рис. 3 видно, что вершина нижней грани является вершиной нижней грани трапецоида, однако соответствующая ей точка верхней грани лежит не на вершине, а на стороне.

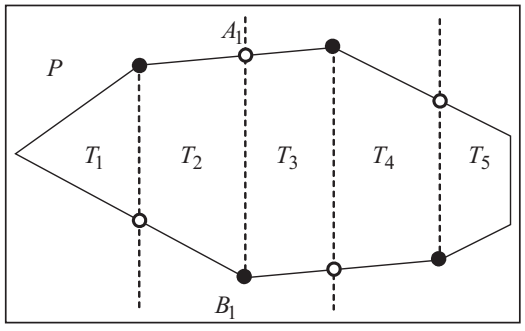

Рис. 3. Разбиение полигона на трапецоиды по верхней и нижней «шапкам»

Для вычисления множества вершин полигона сначала преобразуем каноническую форму элемента (3) следующим образом.

1. Для каждого трапецоида

$$T_i = S_{i1}.T_i^{(n-1)},\ S_{i1} = [L_{i1}, x_n, G_{i1}]$$

выделим его нижнюю и верхнюю шапки:

$$T_i = T_i^+ ++ T_i^-,\ T_i^+ = [-\infty, x_n, G_{i1}].T_i^{(n-1)},\ T_i^- = [L_{i1}, x_n, +\infty].T_i^{(n-1)}.$$

2. Положим $S_{i1}^+ = (x_n, G_{i1}]$, $S_{i1}^- = [L_{i1}, x_n)$, $T_i^+ = S_{i1}^+.T_i^{(n-1)}$, $T_i^- = S_{i1}^-.T_i^{(n-1)}$.

3. Преобразуем форму (3) к виду

$$P = P^+ \& P^- = (T_1^+ ++ T_2^+ ++ \ldots ++ T_k^+) \& (T_1^- ++ T_2^- ++ \ldots ++ T_k^-),$$

$$P^+ = T_1^+ ++ T_2^+ ++ \ldots ++ T_k^+,\ P^- = T_1^- ++ T_2^- ++ \ldots ++ T_k^-.$$

4. Приведем подобные члены в представлениях полигонов P^+, P^-, используя соотношение $S.T_1 ++ S.T_2 = S.(T_1 ++ T_2)$ и свойство дистрибутивности

$$[L, x, H].T ++ [H, x, G].T = [L, x, G].T.$$

В результате получим представление верхней и нижней шапок полигона

$$P^+ = s_1^+.P_1^+ + \ldots + s_k^+.P_{k^+}^+,\ P^- = s_1^-.P_1^- + \ldots + s_k^-.P_{k^+}^-,$$

в котором верхние $(n-1)$ грани полигона P заданы уравнениями s_i^+ вида $x_n = G_i(x_1, \ldots, x_{n-1})$ и ограничены $(n-1)$-мерными полигонами P_i^+, а нижние $(n-1)$ грани полигона P заданы уравнениями s_i^- вида $x_n = L_i(x_1, \ldots, x_{n-1})$ и ограничены $(n-1)$-мерными полигонами P_i^-. Применив рекурсивно сверху–вниз преобразования 1–4 к P_i^+, P_j^-, получим еще одну каноническую форму P, каждая k-я грань которой по существу представлена своими вершинами. Алгоритм преобразования иллюстрируется рис. 4. Назовем эту каноническую форму верхне–нижней нормальной формой (ВННФ).

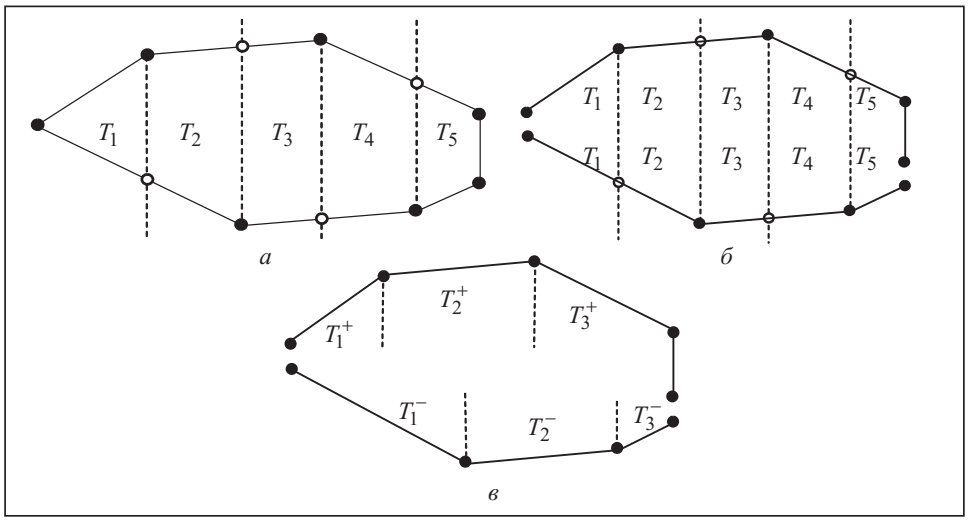

Рис. 4. Разбиение полигона на трапецоиды в форме ПНФ (*а*); отделение верхних и нижних шапок (*б*); преобразование полигона к третьей канонической форме (*в*)

Рассмотрим пример построения ВННФ для полигона, представленного на рис. 5, которое приводит к следующему результату:

$$P^+ = (y = f_1(x)).(x = a_1^-, x = a_1^+) ++$$

$$(y = f_2(x)).(x = a_2^-, x = a_2^+) ++$$

$$(y = f_3(x)).(x = a_3^-, x = a_3^+) + +(x = a_3^+),$$

$$P^- = (y = g_1(x)).(x = b_1^-, x = b_1^+) + +$$

$$(y = g_2(x)).(x = b_2^-, x = b_2^+) + +$$

$$(y = g_3(x)).(x = b_3^-, x = b_3^+).$$

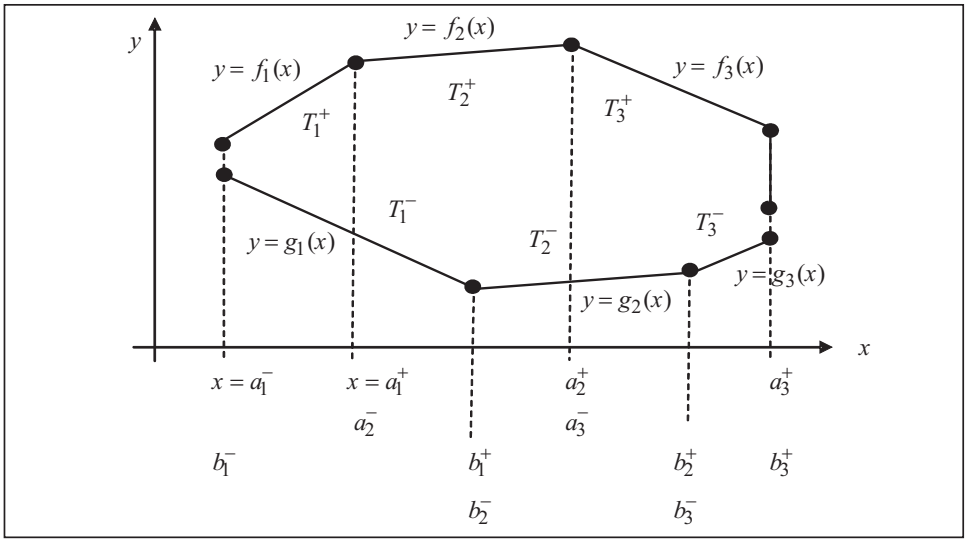

Рис. 5. Иллюстрация алгоритма вычисления вершин полигона

Заметим, что $a_1^+ = a_2^-$, $a_2^+ = a_3^-$, $f_1(a_1^+) = f_2(a_2^-)$, $f_2(a_2^+) = f_3(a_3^-)$, поскольку координаты вершин ломаной — верхней шапки, вычисленные по уравнениям смежных звеньев, совпадают. Аналогичные равенства имеют место и для нижней шапки. Формулы для полигонов P^+, P^- позволяют легко вычислить координаты вершин верхней и нижней шапок снизу–вверх.

Общий алгоритм вычисления базисных вершин. Очевидный и простой алгоритм вычисления базисных вершин полигона (базисных векторов) заключается в следующем.

Вычисления осуществляются снизу–вверх по формуле третьей канонической формы. При этом вычисляются базисы граней проекций трапецоидов в k-мерных подпространствах по k от единицы до n.

1. Если вычислен базис $V^{(k)}(P) = \{v_1, \dots, v_l\}$ и уравнение шапки S имеет вид $x_{k+1} = f(x_1, \dots, x_k)$, то базис $k+1$ грани $S.P$ вычисляется по формуле

$$V^{(k+1)}(S.P) = \{(v_1, f(v_1)), \dots, (v_l, f(v_l))\}.$$

2. Если вычислены базисы $V^{(k)}(P) = \{v_1, \dots, v_l\}$, $V^{(k)}(Q) = \{u_1, \dots, u_r\}$ k-полигонов P, Q, то базис $V^{(k)}(P++Q)$ вычисляется по формуле

$$V^{(k)}(P++Q) = V^{(k)}(P) \cup V^{(k)}(Q).$$

3. Если вычислены базисы $V^{(k)}(P^+) = \{v_1, \dots, v_l\}, V^{(k)}(P^-) = \{u_1, \dots, u_r\}$ k-полигонов P^+, P^-, то базис $V^{(k)}(P^+, P^-)$ вычисляется по формуле

$$V^{(k)}(P^+, P^-) = V^{(k)}(P^+) \cup V^{(k)}(P^-).$$

Базис рекурсивных вычислений очевиден: $V^{(1)}(x_1 = a) = (a)$.

Смежные грани вычисляются аналогично: если вычислены базисы $V^{(k)}(P) = \{v_1, \ldots, v_l\}, V^{(k)}(Q) = \{u_1, \ldots, u_r\}$ k полигонов P, Q, то базис $V^{(k)}(P \cap Q)$ вычисляется по формуле $V^{(k)}(P \cap Q) = V^{(k)}(P) \cap V^{(k)}(Q)$.

МЕТОДЫ ОТЛАДКИ АЛГОРИТМА ПОСТРОЕНИЯ МНОЖЕСТВА ВЕРШИН МНОГОГРАННИКА

Программа построения множества $V(P)$ не может быть отлажена вручную. Поэтому рассмотрим задачу автоматической генерации отладочных примеров.

Требования к программному модулю генерации отладочных примеров.

1. ПМ «Отладчик» должен генерировать примеры в пространствах любой размерности и с любым количеством неравенств.

2. Отладочные примеры должны определяться в виде серий однотипных примеров.

3. Проверка правильности решения примера должна выполняться автоматически (не требовать от пользователя ручных вычислений).

Серия 1. Пусть $C^{(n)}$ — единичный n-мерный куб, заданный множеством единичных векторов, и A — невырожденный линейный (аффинный) оператор, который сохраняет свойство выпуклости n-мерного многогранника. Обозначим $S(C^{(n)})$ систему из n пар линейных неравенств, определяющих $C^{(n)}$; $A(S(C^{(n)}))$ — линейное (аффинное) преобразование системы $S(C^{(n)})$, $A\lg(S)$ — результат применения алгоритма вычисления множества вершин. Правильность алгоритма проверяется соотношением $C^{(n)} = A^{-1}(A\lg(A(S(C^{(n)}))))$.

Серия 2. Пусть $V = \{V_1, \ldots, V_{n+1}\}$ — множество векторов n-мерного пространства, $S(V)$ — система, состоящая из $n+1$ линейного неравенства, определяющая n-мерный гипертетраэдр через его n граней. Правильность алгоритма проверяется соотношением $V = A\lg(S(V))$.

Реализация серии требует алгоритма построения уравнения гиперплоскости — n-грани, проходящей через n точек пространства. Пусть $l(X) = 0$ — такое уравнение. Тогда искомое ЛН имеет вид $l(X) \le 0$ или $l(X) \ge 0$. Чтобы получить ограниченный гипертетраэдр, достаточно вычислить центр тяжести системы точек векторов $V = \{V_1, \ldots, V_{n+1}\}$:

$$V_m = \frac{1}{n+1}(V_1 + \ldots + V_{n+1}).$$

Знак числа $l(V_m)$ совпадает с искомым знаком неравенства.

Серии 1, 2 создают примеры, в которых каждое неравенство является основным. Для получения неравенств — следствий из системы основных неравенств, используется следующий очевидный факт.

Пусть $S = (l_1(X) \ge 0) \& \ldots (l_k(X) \ge 0)$ — СЛН и $a_1, \ldots, a_k, a_{k+1}$ — неотрицательные числа. Тогда $l(X) = a_1 l_1(X) + \ldots + a_k l_k(X) + a_{k+1}$ — следствие системы S. Если система S содержит только основные неравенства и $a_{k+1} = 0$, то это следствие является строгим. Если система S содержит только основные неравенства и $a_{k+1} < 0$, неравенство $l(X)$ является основным.

Серии 1, 2 могут быть дополнены алгоритмом генерации следствий из основных систем этих серий. Входные системы в этом случае содержат как основные неравенства, так и следствия из них, расположенные в системе хаотически.

Кроме того, систему можно дополнить несколькими основными неравенствами, «отрезая» от основного геометрического тела некоторую часть с легко вычисляемыми вершинами. Например, от трехмерного единичного куба можно «отрезать» вершину $(1,1,1)$, заменив ее тремя вершинами: $(2/3,1,1)$, $(1,2/3,1)$, $(1,1,2/3)$.

РЕАЛИЗАЦИЯ АЛГОРИТМА СРЕДСТВАМИ ИНСЕРЦИОННОГО МОДЕЛИРОВАНИЯ

Для реализации описанных выше алгоритмов используем систему инсерционного моделирования IMS [12], созданную на основе системы алгебраического программирования APS [13] под руководством академика НАНУ А.А. Летичевского.

Система алгебраического программирования APS разработана в середине 80-х годов в Институте кибернетики НАН Украины (Киев) [14]. В 2010 г. на базе системы APS была создана система инсерционного моделирования IMS в рамках сотрудничества между Институтом кибернетики и Херсонским государственный университетом [15].

Системы APS и IMS применяются для прототипирования алгоритмов и моделирования программных систем. Они успешно применяются в ряде коммерческих продуктов: VRS (Verification for Requirement Specification) [16], TerM [17], что позволяет авторам APS и IMS позиционировать их как коммерческие системы [18].

Инсерционное моделирование — направление, развиваемое в последнее десятилетие как подход к исследованию распределенных многоагентных систем и разработке средств верификации распределенных параллельных программ и аппаратуры [12].

Общепринятым средством для описания динамики систем в современной компьютерной науке являются транзиционные системы, которые определяются множеством состояний и отношением переходов. Обычно это понятие обогащается различными дополнительными структурами, важнейшей из которых является разметка переходов (размеченные транзиционные системы, введенные Д. Парком [19] для описания поведения автоматов на бесконечных словах). В инсерционном моделировании исходным является понятие атрибутной транзиционной системы [12], которое формально определяется как пятерка $< S, A, U, T, \varphi >$ (где S — множество состояний, A — множество действий, используемых для разметки переходов, U — множество атрибутных разметок, используемых для разметки состояний, T — отношение переходов, $T \subset S \times A \times S \cup S \times S$) и состоит из размеченных переходов $s \xrightarrow{a} s'$ и неразмеченных переходов $s \to s'$. Эта часть структуры соответствует обычному понятию размеченной транзиционной системы (со скрытыми переходами). Функция $\varphi : S \to U$ называется функцией разметки состояний. Обычно U определяется как множество $U = D^R$ отображений множества R атрибутов во множество данных D (область значений атрибутов) или в случае типизированных данных как семейство $U = (D_\xi^{R_\xi})_{\xi \in \Xi}$, где Ξ — множество типов данных. Обычно это интерпретированный или неинтерпретированный язык первого порядка, возможно, расширенный некоторыми модальностями темпоральной логики.

Транзиционные системы могут также настраиваться путем выделения некоторых специфических множеств состояний из множества состояний S. Важнейшими из них являются множества начальных S_0, заключительных S_Δ и неопределенных $S_\perp$ состояний. Последние используются в теории для определения отношения аппроксимации и построения бесконечных систем в виде пределов конечных.

В инсерционном моделировании в качестве инвариантов используются (в общем случае бесконечные) выражения или системы уравнений в алгебре поведений. Сама алгебра поведений устроена достаточно просто. Она представляет собой двухосновную алгебру $< U, A >$, первый компонент которой — множество U поведений, а второй — множество действий A. Сигнатура алгебры поведений

состоит из двух операций, одного отношения и трех констант. Первая операция $a.u$ называется префиксингом. Ее аргументами являются a — действие и u — поведение, а результатом — новое поведение. Вторая операция — недетерминированный выбор $u+v$. Это бинарная операция, которая определена на множестве поведений. Она коммутативна, ассоциативна и идемпотентна. Константы алгебры поведений — успешное завершение Δ, неопределенное поведение $\perp$ и тупиковое поведение 0, которое является нейтральным элементом недетерминированного выбора. На множестве поведений определено бинарное отношение аппроксимации $\subseteq$, которое является отношением частичного порядка с наименьшим элементом $\perp$. Операции префиксинга и недетерминированного выбора монотонны и непрерывны относительно этого отношения. В основной роли выступает полная алгебра поведений $F(A)$, которая содержит пределы всех направленных множеств. Следовательно, в этой алгебре имеет место теорема о наименьшей неподвижной точке. Точная конструкция алгебры $F(A)$ (для произвольного, включая бесконечное, множества действий) изложена в [20].

Спроектируем описанные выше алгебры в терминах расширений (двойная стрелка) и наследования (одинарная стрелка) (рис. 6) [4, 6, 21]. Здесь *LinMonom* — одномерное векторное пространство над полем *Coef*, $\Sigma = \{++,-,*,/,<,>,==\}$ (деление и умножение на число) и носителем $\mathbf{S} = \{c* v, v \in Variable, c \in Coef, c \neq 0\}$.

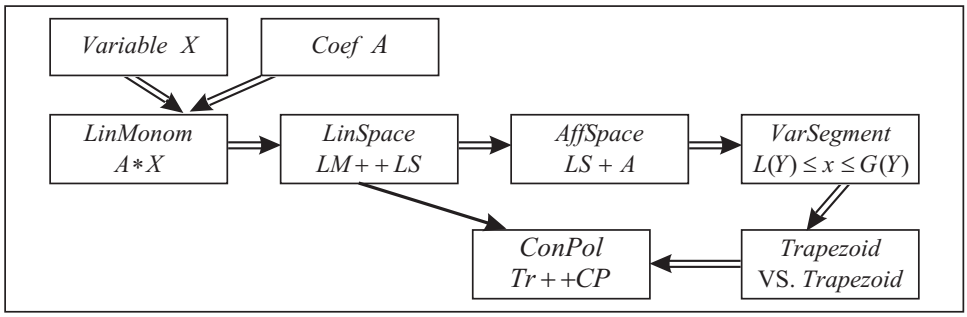

Рис. 6. Диаграмма расширений и наследования сортов

LineSpace — векторное пространство линейных комбинаций над полем *Coef*, $\Sigma = \{++,-,*,/,==\}$(деление и умножение на число) и носителем $\mathbf{S} = \{lm + ls, lm \in \in LinMonom, ls \in LineSpace\}$.

Affspace — аффинное пространство линейных комбинаций со свободным членом над полем *Coef*, $\Sigma = \{++,-,*,/, NOD, NOK<,>,==\}$ (деление и умножение на число) и носителем $\mathbf{S} = \{ls++c, ls \in LinSpace, c \in Coef\}$.

Расширение алгебры происходит за счет введения новых конструкторов элементов носителей, описанных ваше, операция пересечения трапецоидов определяет наследование между сортами *LinSpace* и *ConPol*.

Для реализации такого алгоритма средствами инсерционного моделирования необходимо реализовать функцию развертывания состояний (unfolding) и функцию погружения агентов в среду (insertion) [20]. Вычисления в алгебре *LinSpace* легко реализовать с помощью оператора языка APLAN *mrg* [21] (операция ++ определяется аналогично операции +), поэтому в дальнейшем будем считать, что в системе линейных неравенств все неравенства приведены к виду $f_k(x_n, \ldots, x_1) \leq 0$.

В качестве начального состояния выберем всю область допустимых решений. Для этого найдем список всех переменных системы неравенств и приведем эту область к носителю алгебры *Trapezoid*: $(-\infty, x_n, +\infty).(-\infty, x_{n-1}, +\infty)$. … Квадратные скобки будем использовать только для обозначения вхождения конца отрезка в промежуток.

Функция развертывания состояний для этого примера определяется как конструктор сорта *VarSegment*:

$unfold\_rs := rs(x, y)($

$(x \le 0) = SolveLinUnEq(x \le 0),$

$(x < 0) = SolveLinUnEq(x < 0)$

);

Функция *SolveLinUnEq* реализует конструктор сорта *LinUnEqu*.

Функция погружения содержит последовательное рассмотрение каждого ограничения и операцию рекурсивного пересечения ограничений с трапецоидом:

$ins := rs(E, P, x, y)($

$$E[Delta] = Delta, \tag{4}$$

$$E[x\&\ y] = ins(ins\,E[x])[y], \tag{5}$$

$$(x + + y)[P] = ins\,x[P] + + ins\,y[P], \tag{6}$$

$$(x.\ y)[P] = split\_tr(ins\,x[P].\ ins\,y[P]), \tag{7}$$

$$E[(x < y)] = cross\_vs(x < y, E, P), \tag{8}$$

$$E[(x \le y)] = cross\_vs(x \le y, E, P), \tag{9}$$

$$E[P] = E[short\_intensP] \tag{10}$$

);

Правило (4) означает, что погружение терминального состояния *Delta* заканчивается формированием трассы «успешное завершение»; (5) — реализация последовательного рассмотрения неравенств; (6), (7) — пересечение ограничения с выпуклым многогранником и трапецоидом соответственно; (8), (9) — пересечение ограничения с элементом сорта *VarSegment* — функция *cross_vs*; (10) — вызов функции развертывания состояний. Функция *split_tr* реализует закон дистрибутивности: $x.(y + + z) = x.\ y + + x.\ z$.

Для входных данных программы используется терм $1[L_1 \& L_2 \ldots \& L_n]$ (L_j — линейное неравенство), тогда функция *cross_vs* при $E = 1$ приводит состояние E к носителю сорта *Trapezoid*.

Таким образом, рассмотрены алгоритм решения системы линейных неравенств и его реализация средствами инсерционного моделирования. Изложенный алгоритм успешно использован для частичной конкретизации символьных трасс.

Авторы благодарят академика А.А. Летичевского за внимание к постановке задачи построения алгоритма решения систем линейных неравенств методами алгебраического программирования.

СПИСОК ЛИТЕРАТУРЫ

1. Львов М.С. Алгебраический подход к задаче решения систем линейных неравенств // Кибернетика и системний анализ. — 2010. — № 2. — С. 175–188.
2. Goguen J., Meseguer J. Ordered-sorted algebra I: Partial and Overloaded Operations. Errors and Inheritance. Computer Science. — Oxford: Elsevier. — 1992. — **105**, N 2. — P. 217–273.
3. Goguen J.A., Thatcher J.W., Wagner E. An initial algebra approach to the specification, correctness and implementation of abstract data types // Current Trends in Programming Methodology (R.Yeh ed.). — NJ: Prentice Hall, 1978. — P. 80–149.
4. Львов М.С. Синтез інтерпретаторів алгебраїчних операцій в розширеннях багатосортних алгебр // Вісн. Харк. нац. ун-ту. — 2009. — № 847. — С. 221–238.
5. Львов М.С. Верифікація інтерпретаторів алгебраїчних операцій в розширеннях багатосортних алгебр. — Харків: ХУПС. — 2009. — Вип. 3 (21). — С. 127–137.

6. Львов М.С. Метод спадкування при реалізації алгебраїчних обчислень в математичних системах навчального призначення // Системи управління, навігації та зв'язку. — К.: ЦНДІНіУ. — 2009. — Вип. 3 (11). — С. 120–130.
7. Львов М.С. Метод морфізмів реалізації алгебраїчних обчислень в математичних системах навчального призначення // Системи обробки інформації. — Харків: ХУПС. — 2009. — Вип. 6 (80). — С. 183–190.
8. Львов М.С. Об одном подходе к реализации алгебраических вычислений: вычисления в алгебре высказываний // Вістн. Харк. нац. ун-ту. — 2009. — № 863. — С. 157–168. (Математическое моделирование. Информационные технологии.)
9. Моцкин Т.С., Райфа Х., Томпсон Дж.Л., Тролл Р.М. Метод двойного описания. Матричные игры. — М.: Физматгиз, 1961. — С. 81–109.
10. Zeidler G.L. Lectures on convex polytopes. — Berlin; New York: Springer-Verlag, 1994. — 152 p.
11. Черников С.Н. Линейные неравенства. — М.: Наука, 1968. — 490 с.
12. Letichevsky A., Kapitonova J., Volkov V., Vyshemirskii V., Letichevsky A.,Jr. Insertion programming // Cybernetics and System Analysis. — 2003. — N 1. — P. 19–32.
13. Letichevsky A.A., Kapitonova J.V., Konozenko S.V. Algebraic programming system APS-1 // Informatics'-89: Proc. of the Soviet-Franch symp. — Tallin, 1989. — P. 46–55.
14. Інститут кібернетики імені В.М. Глушкова НАНУ [Электронный ресурс]/ИК.— Электрон. Дан, 2011, Режим доступа: http//www.icyb.kiev.ua, свободный. — Загл. с экрана. — Яз. укр., рус., англ.
15. Херсонский государственный университет [Электронный ресурс]/ХГУ. — Электрон. Дан, 2010, Режим доступа: http//www.ksu.ks.ua, свободный. — Загл. с экрана. — Яз. укр., рус., англ.
16. Летичевский А.А., Капитонова Ю.В., Волков В.Ф. и др. Сертификация систем с помощью базовых протоколов // Кибернетика и системный анализ. — 2005. — № 4. — С. 256–268.
17. Программно-методический комплекс Терм VII поддержки практической математической деятельности [Электронный ресурс] /ХГУ. — Электрон. Дан, 2011, Режим доступа: http://riit.ksu.ks.ua/index.php?q=ru/node/229, свободный. — Загл. с экрана. — Яз. рус.
18. APS & IMS [Электронный ресурс]/ЛитСофт.— Электрон. Дан, 2011, Режим доступа: http//www.apsystem.org,ua, свободный. — Загл. с экрана. — Яз. англ.
19. Park D. Concurrency and automata on infinite sequences. // LNCS. — 1981. — 104. — P. 167–183.
20. Letichevsky A. Algebra of behavior transformations and its applications / V.B. Kudryavtsev and I.G. Rosenberg eds // Structural theory of Automata, Semigroups, and Universal Algebra, NATO Science Series II. Mathematics, Physics and Chemistry. — 2005. — **207**. — P. 241–272.
21. Песчаненко В.С. Об одном подходе к проектированию алгебраических типов данных // Проблеми програмування. — 2006. — № 2–3. — С. 626–634.

Поступила 22.09.2011