УДК 519.8+004.416.3+004.627

В.И. ШИНКАРЕНКО, Т.Н. ВАСЕЦКАЯ

МОДЕЛИРОВАНИЕ ПРОЦЕССА АДАПТАЦИИ АЛГОРИТМОВ СЖАТИЯ СРЕДСТВАМИ КОНСТРУКТИВНО-ПРОДУКЦИОННЫХ СТРУКТУР

Аннотация. Представлена методология математико-алгоритмического конструктивизма. Определены связи конструктивно-продукционных структур, моделирующих взаимосвязанные конструкции и конструктивные процессы. Разработана модель процесса адаптации алгоритмов сжатия в виде конструктивно-продукционных структур: двойственного конструктора алгоритмов сжатия и декомпрессии, преобразователя алгоритма сжатия в конструктивный процесс сжатия и адаптера.

Ключевые слова: сжатие, декомпрессия, адаптация, конструктивно-продукционная структура, моделирование, конструктор, адаптер, преобразователь.

ВВЕДЕНИЕ

Для моделирования и исследования программ и алгоритмов применяются средства алгоритмических алгебр, схем программ, грамматик и автоматов, машин Тьюринга и др. [1, 2].

В работах [3–5] заложены основы математико-алгоритмического конструктивизма (МАК). С использованием конструктивно-продукционных структур (КПС), как основного средства МАК, можно моделировать любые конструкции и конструктивные процессы в области информационных технологий, строительства, машиностроения, робототехники, биологии и т.д. Предложенный аппарат позволяет формализовать процессы и результаты формирования конструкций различной природы, связывая элементы конструкций и учитывая свойства элементов, их агрегатов (форм) и связей.

Основой моделирования в МАК есть обобщенная конструктивно-продукционная структура (ОКПС) [3]. Ее особенностями являются: атрибутивность элементов и операций, расширяемый носитель, модель исполнителя в виде его базовых алгоритмов, связь операций с алгоритмами их выполнения. Модели конкретных конструкций или процессов формируются с помощью КПС в результате определения их специализации, интерпретации, конкретизации ОКПС и реализации.

В данной работе развивается методология МАК для формирования и взаимодействия нескольких взаимосвязанных конструкций и конструктивных процессов, а в задаче адаптации алгоритмов сжатия — одновременного формирования прямого и обратного конструктивных процессов. (Сжатие (архивация, компрессия) — уменьшение объема, занимаемого данными [6], обратный процесс — декомпрессия, разархивация.) В работе [7] представлен подход к адаптации алгоритмов сжатия на основе метаалгоритмов (обобщение на основе КПС).

Каждая КПС имеет определенное предназначение. Для моделирования процесса адаптации алгоритмов сжатия выделены следующие КПС: конструктор, преобразователь и адаптер. Конструктор формирует допустимые конструкции, в данном случае — алгоритмы сжатия. Преобразователь сконструированный алгоритм преобразует в процесс сжатия. Адаптер содержит средства для работы с атрибутами, по которым выполняется адаптация. В настоящей статье принят критерий качества адаптации — степень сжатия.

ОБОБЩЕННАЯ КОНСТРУКТИВНО-ПРОДУКЦИОННАЯ СТРУКТУРА

Рассмотрим ОКПС, определенную в [3] как

$$C_G = \langle M, \Sigma, \Lambda \rangle,$$

где M — неоднородный носитель структуры; Σ — сигнатура, состоящая из множеств операций связывания, подстановки и вывода, операций над атрибутами, а также из отношения подстановки; Λ — конструктивная аксиоматика [3].

Назначение КПС состоит в формировании множеств конструкций с помощью операций связывания, подстановки, вывода и других операций, заданных правилами аксиоматики. Для формирования конструкций необходимо выполнять ряд уточняющих преобразований ОКПС:

- определить специализацию предметной области, т.е. семантическую природу носителя, конечное множество операций и их семантику, атрибутику операций, порядок их выполнения и ограничения на правила подстановки;
- осуществить интерпретацию, т.е. связать операции сигнатуры с алгоритмами построения некоторой алгоритмической структуры их выполнения [4], при этом связать информационную модель способа построения конструкций с моделью исполнителя;
- конкретизировать ОКПС, т.е. расширить аксиоматику множеством правил продукций, задать конкретные множества нетерминальных и терминальных символов с их атрибутами и при необходимости — значения последних;
- реализовать ОКПС, т.е. сформировать конструкции из элементов носителя КПС путем выполнения алгоритмов, связанных с операциями сигнатуры (реализация возможна только для предварительно специализированной, интерпретированной и конкретизированной КПС [3]).

СПЕЦИАЛИЗАЦИЯ ДВОЙСТВЕННОГО КОНСТРУКТОРА АЛГОРИТМОВ СЖАТИЯ

Выполнив специализацию ОКПС, определим двойственный конструктор алгоритмов сжатия без потерь, одновременно формирующий алгоритмы сжатия и декомпрессии

$$C = \langle M, \Sigma, \Lambda \rangle_S \mapsto C_{KAC} \langle M_{KAC}, \Sigma_{KAC}, \Lambda_{KAC} \rangle,$$

где $\Lambda_{KAC} = \Lambda \cup \Lambda_1 \cup \Lambda_2$, $\Lambda_1 = \{M_{KAC} \supset T_1 \cup N_1$, $\Sigma_{KAC} = \{\Xi, \Theta, \Phi\}$, $\Xi = \{\cdot, :\}$, $\Phi = \{+, :=, =, \neq, \div, \otimes, \vee\}$, $\Theta = \{\Rightarrow, |\Rightarrow, ||\Rightarrow\}\}$.

Частичная аксиоматика Λ_2 содержит определения, дополнения и ограничения, которые уточняют элементы носителя с их атрибутами, отношения подстановки, задают особенности выполнения операций подстановки и вывода. Рассмотрим их.

1. Обратный алгоритм $A^{-1}|_Y^X$, для которого справедливо следующее (в нотации, предложенной в [4]):

$$A|_X^Y \cdot A^{-1}|_Y^X = E|_X^X,$$

где $E|_X^Y$ — единичный алгоритм, реализующий функцию $y = x$, $x \in X$, $y \in Y$.

2. Нетерминальный алфавит $N_1 = \{{}_{\varphi}\alpha_i\}$ состоит из множества абстрактных алгоритмов, где $\varphi \leftarrow\!\shortmid \alpha$ — семантика. Терминальный алфавит $T_1 = \{O = \{{}_{p_i}B_i^0\}$, $\breve{O} = \{{}_{p_i}(B_i^{-1})^0\}, E|_X^Y\}$ содержит конечное множество базовых (образующих) алгоритмов ${}_{p_i}B_i^0$ и множество соответствующих им обратных алгоритмов $(B_i^{-1})^0$, $p_i \leftarrow\!\shortmid B_i^0$ — рекомендуемая вероятность применения i-го алгоритма. Образующий алгоритм реализует атомарные действия (считающиеся неделимыми). Множество базовых алгоритмов — это множество образующих алгоритмов некоторого исполнителя [4].

3. Неоднородный носитель M_{KAC} содержит образующие и конструируемые [8] алгоритмы, а также необходимые для них данные.

Согласно аксиоматике ОКПС формой ${}_{w}l$ с атрибутом w называется набор терминалов и нетерминалов, объединенных операциями связывания, а конструкцией — форма, содержащая только терминалы [3].

4. Правило подстановки имеет вид ψ_r: $\langle s_r, g_r\rangle \in \Psi$, где $s_r = \langle \bar{s}_r, \tilde{s}_r\rangle$ — два отношения подстановки: $\bar{s}_r$ — для алгоритма сжатия, $\tilde{s}_r$ — для алгоритма декомпрессии; g_r — набор операций над атрибутами. Отношение подстановки является двухместным с атрибутами ${}_{w_i} l_i \; {}_{w}\!\rightarrow {}_{w_j} l_j$ [3]. Для формы ${}_{w_l} l_l = {}_{w_0} \oplus ({}_{w_1} l_1, {}_{w_2} l_2, \ldots, {}_{w_h} l_h, \ldots, {}_{w_k} l_k)$ и доступного отношения подстановки ${}_{w_h} l_h \; {}_{w_p}\!\rightarrow {}_{w_q} l_q$ такого, что ${}_{w_h} l_h \prec {}_{w_l} l_l$ (${}_{w_h} l_h$ — часть ${}_{w_l} l_l$), результатом трехместной операции подстановки ${}_{w_p}\Rightarrow ({}_{w_h} l_h, {}_{w_q} l_q, {}_{w_l} l_l)$ будет форма ${}_{w_l^*} l_l^* = {}_{w_0} \oplus ({}_{w_1} l_1, {}_{w_2} l_2, \ldots, {}_{w_q} l_q, \ldots, {}_{w_k} l_k)$, где $\oplus$ — любая операция связывания из Ξ [3].

5. Двухместная операция частичного вывода ${}_{maxn, \vec{q}, m} l^* = (|\Rightarrow (\Psi, {}_{maxn, \vec{q}, m} l))$, где ${}_{maxn, \vec{q}, m} l$ и ${}_{maxn, \vec{q}, m} l^*$ — формы соответственно до и после выполнения операции подстановки, $\vec{q}$ — вектор количества вхождений базовых алгоритмов ${}_{p_i} B_i^0 \in T_1$ в конструируемый, m — общее количество базовых алгоритмов в форме, $maxn$ — максимально допустимое количество базовых алгоритмов в форме, заключается в следующем:

- выбирается одно из доступных правил подстановки ${}_{\vec{d}}\psi_r : \langle s_r, g_r\rangle \in \Psi$ с отношениями подстановки s_r и на его основе выполняются операции подстановки, где $\vec{d}$ — вектор доступности правила (r-е правило доступно, если $d_r = 1$, и не доступно, если $d_r = 0$);
- выполняются операции подстановки на основе отношения подстановки $\tilde{s}_r$ для формирования обратного алгоритма;
- выполняются операции над атрибутами g_r.

6. Порядок применения операции над атрибутами в процессе выполнения операции частичного вывода задается ее атрибутом τ_j, $\tau_j \in I$, где $I = \{\tau_0, \tau_1\}$, $I \subset M_{KAC}$, τ_0 и τ_1 — определяют, что операции над атрибутом должны выполняться соответственно до и после операции подстановки.

7. Операция полного вывода (или вывода) заключается в последовательном выполнении операции частичного вывода, начиная с начального нетерминала и заканчивая конструкцией, удовлетворяющей условию окончания вывода, в результате чего создается пара конструкций: алгоритм сжатия и декомпрессии.

8. Условием окончания вывода является отсутствие нетерминалов в форме.

9. Вводятся следующие операции над атрибутами:

- операция выбора $\otimes (p; k; \bar{L}; \bar{\bar{L}})$, результатом которой является i-й элемент списка $\bar{L}$, если $p \in (a_i, b_i)$, с операндами $\bar{L}$ — список k целых чисел, $\bar{\bar{L}}$ — список k отрезков (a_i, b_i) таких, что $a_1 = 0$, $b_k = 1$, $a_{i+1} = b_i$, p — число в интервале $[0, 1]$);
- операция $\div (c; n; L)$ заключается в выполнении n операций, представленных в префиксной форме, из списка L, $L = (j_1, j_2, \ldots, j_n)$, если $c = true$;
- операция $\vee (a, b)$, результатом которой является случайное число на промежутке $[a, b]$.

ИНТЕРПРЕТАЦИЯ ДВОЙСТВЕННОГО КОНСТРУКТОРА АЛГОРИТМОВ СЖАТИЯ

Для интерпретации специализированной ОКПС необходимо задать базовую алгоритмическую структуру (БАС) [4, 9] и связать ее алгоритмы с операциями сигнатуры специализированной КПС конструктора алгоритмов сжатия.

Пусть имеется следующая БАС:

$$C_{A,KAC} = \langle M_{A,KAC}, V_{A,KAC}, \Sigma_{A,KAC}, \Lambda_{A,KAC} \rangle,$$

где $M_{A,KAC}$ — неоднородный носитель, $\Sigma_{A,KAC}$ — сигнатура, $\Lambda_{A,KAC}$ — аксиоматика, $V_{A,KAC} \supset \{A_1^0|_{A_i,A_j}^{A_i \cdot A_j}, A_2^0|_{Z_1,Z_2,A_i}^{:A_i}\}$ — множество образующих алгоритмов для некоторого исполнителя. В БАС имеется множество сконструированных алгоритмов $\{A_3|_{l_h,l_q,f_i}^{f_j}, A_4|_{f_i,\Psi}^{f_j}, A_5|_{f_i,\Psi}^{f_j}\} \bigcup V_w$ и $\{A_6^0|_a^a, A_7^0|_a^b, A_8^0|_{a,b}^p, A_9|_{p,k,L_1,L_2}^c, A_{10}^0|_{a,b}^c, A_{11}^0|_{a,b}^c, A_{12}|_{c,n,L}^L\} \in V_w$, реализующих операции Ф над атрибутами.

Данные алгоритмы реализуют следующие операции:

- $A_1^0|_{A_i,A_j}^{A_i \cdot A_j}$ — конкатенация алгоритмов (последовательное выполнение алгоритма A_i после A_j);
- $A_2^0|_{Z_1,Z_2,A_i}^{:A_i}$ — условное выполнение алгоритма A_i, т.е. если $Z_1 \in Z_2$, то алгоритм A_i выполняется, в противном случае не выполняется;
- $A_3|_{l_h,l_q,f_i}^{f_j}$ — подстановка;
- $A_4|_{f_i,\Psi}^{f_j}, A_5|_{\sigma,\Psi}^{\overline{\Omega}}$ — частичный и полный вывод, где f_i, f_j — формы, σ — начальный нетерминал, $\overline{\Omega}$ — множество сформированных конструкций;
- $A_6^0|_a^a$ — инкремент a;
- $A_7^0|_a^b$ — присваивание $b := a$, где b и a могут являться числами, массивами или списками;
- $A_8^0|_{a,b}^p$ — генерация случайного числа p из промежутка от a до b;
- $A_9|_{p,k,\overline{L},\overline{\overline{L}}}^c$ — многоместная операция выбора $\otimes$;
- $A_{10}^0|_{a,b}^c$ — проверка неравенства a и b, т.е. c принимает значение *true*, если $a \neq b$, и *false* в противном случае;
- $A_{11}^0|_{a,b}^c$ — проверка равенства a и b, т.е. c принимает значение *true*, если $a = b$, и *false* в противном случае;
- $A_{12}|_{c,n,L}^L$ — выполнение n алгоритмов из списка L, если $c = true$.

Рассмотрим интерпретацию КПС для двойственного конструктора алгоритмов сжатия

$$\langle C_{KAC} = \langle M_{KAC}, \Sigma_{KAC}, \Lambda_{KAC} \rangle, C_{A,KAC} =$$
$$= \langle M_{A,KAC}, V_{A,KAC}, \Sigma_{A,KAC}, \Lambda_{A,KAC} \rangle \rangle_I \mapsto$$
$$_I\mapsto C_{KAC} = \langle M_{KAC}, \Sigma_{KAC}, \Lambda_{I,KAC} \rangle,$$

где $_I\mapsto$ — операция интерпретации; $\Lambda_{I,KAC} = \Lambda_{KAC} \bigcup \Lambda_3$, $\Lambda_3 = \{(A_1^0|_{A_i,A_j}^{A_i \cdot A_j} \hookleftarrow \cdot), (A_2^0|_{Z_1,Z_2,A_i}^{A_i} \hookleftarrow :), (A_3|_{l_h,l_q,f_i}^{f_j} \hookleftarrow \Rightarrow), (A_4|_{f_i,\Psi}^{f_j} \hookleftarrow |\Rightarrow), A_5|_{\sigma,\Psi}^{\overline{\Omega}} \hookleftarrow ||\Rightarrow), (A_6^0|_a^a \hookleftarrow +), (A_7^0|_a^b \hookleftarrow :=), (A_8^0|_{a,b}^p \hookleftarrow \vee), (A_9|_{p,k,\overline{L},\overline{\overline{L}}}^c \hookleftarrow \otimes), (A_{10}^0|_{a,b}^c \hookleftarrow \neq), (A_{11}^0|_{a,b}^c \hookleftarrow =), (A_{12}|_{c,n,L}^L \hookleftarrow \div)$.

Для конкретизации интерпретированной структуры ${}_{I,C_{A,KAC}}C_{KAC}$ определим базовые алгоритмы сжатия, на основе которых будет выполняться адаптация. Рассмотрим конкретизированную структуру ${}_{I,C_{A,KAC}}C_{KAC}$:

$$ {}_{I,C_{A,KAC}}C_{KAC} = \langle M_{KAC}, \Sigma_{KAC}, \Lambda_{I,KAC} \rangle_K \mapsto C_{K,KAC} = $$
$$ = \langle M_{KAC}, \Sigma_{K,KAC}, \Lambda_{I,KAC} \cup \Lambda_4 \cup \Lambda_5 \rangle, $$

где $\Lambda_4 = \{T_1 = \{O = \{B_1^0|_M^{Ms}, B_2^0|_M^{Ms}, B_3^0|_M^{M}, B_4^0|_M^{Ms}, \{B_i^0|_M^{Ms}\ \forall i = 5,\dots,8\};\ B_9^0|_{M,fs}^{fs},$ $B_{10}^0|_{M,Md,Mf,s}^{Mc},\ B_{11}^0|_{Ma,Mb}^{Ma,Mb},\ B_{12}^0|_M^{Ma},\ B_{13}^0|_{Ma}^{Mb,Af,Bf},\ B_{14}^0|_{Ma,Mb}^{Ma,Mb},\ B_{15}^0|_{Ma}^{Mc},\ B_{16}^0|_{Ma}^{Mb},$ $B_{17}^0|_f^M\},\quad \breve{O} = \{(B_1^{-1})^0|_{Ms}^M, (B_2^{-1})^0|_{Ms}^M, (B_3^{-1})^0|_M^M,\quad (B_4^{-1})^0|_{Ms}^M,\quad \{(B_i^{-1})^0|_{Ms}^M$ $\forall i = 5,\dots,8\};\quad (B_9^{-1})^0|_{fs}^{M,fs},\quad (B_{10}^{-1})^0|_{Mc}^{M,Md,Mf,s},\quad (B_{11}^{-1})^0|_{Ma,Mb}^{Ma,Mb},\quad (B_{12}^{-1})^0|_{Ma}^{M},$ $(B_{13}^{-1})^0|_{Mb,Af,Bf}^{Ma},\ (B_{14}^{-1})^0|_{Ma,Mb}^{Ma,Mb},\ (B_{15}^{-1})^0|_{Mc}^{Ma},\ (B_{16}^{-1})^0|_{Mb}^{Ma}, (B_{17}^{-1})^0|_f^M\}\},\ N_1 = \{\sigma|_M^{Ms},$ $\alpha_1|_M^{Ms},\ \alpha_2|_M^{Ms}, \alpha_3|_M^{Ms}, \alpha_4|_M^{Ms},\ \alpha_5|_M^{Md,Mf,s}, \alpha_6|_{Mc}^{Mb}, \alpha_7|_M^{Ms},\ \beta_1|_{Ms}^M,\ \beta_2|_{Ms}^M,\ \beta_3|_{Ms}^M,$ $\beta_4|_{Ms}^M,\ \beta_5|_{Md,Mf,s}^M, \beta_6|_{Mb}^{Mc}, \beta_7|_{Ms}^M\},\ U = \{\sigma\},\ \Psi_K = \{{}_{\bar{d}}\psi_i : \langle s_i, g_i \rangle\},\ i = 1,\dots,9\}$, здесь U — множество начальных нетерминалов.

Далее представлены базовые алгоритмы B_i^0, примененные в [7, 10]:

- $B_1^0|_M^{Ms}$, $B_2^0|_M^{Ms}$, $B_4^0|_M^{Ms}$ — предварительная обработка arcDelta2 [11], arcBase64 [12] и RLE соответственно;
- $B_3^0|_M^M = E|_M^M$ — пустой проход;
- $B_5^0|_M^{Ms}$, $B_6^0|_M^{Ms}$, $B_7^0|_M^{Ms}$, $B_8^0|_M^{Ms}$ — сжатие методом arcDeflate [13, с. 94], arcBZip, arcLZMA [13, с. 90] и GZip соответственно, где M — массив для сохранения сжимаемых данных;
- $B_9^0|_M^{fs}$ — сохранение данных в файл с именем f_s;
- $B_{10}^0|_{M,Md,Mf,s}^{Mc}$, $B_{11}^0|_{Ma,Mb}^{Ma,Mb}$, $B_{12}^0|_M^{Ma}$, $B_{13}^0|_{Ma}^{Mb,Md,Mf}$, $B_{14}^0|_{Ma,Mb}^{Ma,Mb}$, $B_{15}^0|_{Ma}^{Mc}$ — алгоритмы для реализации сжатия методом арифметического кодирования (кодирование массива, обнуление элементов входных массивов *Ma* и *Mb*, определение частоты символов, подсчет накапливаемых частот символов массива, «быстрая» сортировка массивов и формирование массива кодов соответственно);
- $B_{16}^0|_{Ma}^{Mb}$ — копирование элементов входного массива в *Mb*;
- $B_{17}^0|_f^M$ — запись содержимого файла(ов) f в масив M.

Множество $\{(B_i^{-1})^0|_{Y_i}^{X_i}\}\ \forall i = 1,\dots,17$ содержит алгоритмы, обратные соответствующим базовым $\{B_i^0|_{X_i}^{Y_i}\}\ \forall i = 1,\dots,17$. Запись $(B_i^{-1})^0$ означает, что обратный алгоритм B_i^{-1} также является образующим.

Символы нетерминального алфавита используются для обозначений абстрактных алгоритмов, которые в процессе вывода заменяются конкретными базовыми алгоритмами. В табл. 1 представлены значения атрибута семантики для символов нетерминального алфавита.

Частичная аксиоматика Λ_5 содержит следующие конструктивные дополнения:

- для элементов вектора вероятностей применения базовых алгоритмов должны выполняться правила $p_1 + p_3 = 1,\ p_2 + p_3 + p_4 = 1,\ \sum_{i=5}^{9} p_i = 1,\ p_{10} = 0.5$;

Таблица 1

Атрибут	Семантика абстрактных алгоритмов	Атрибут	Семантика абстрактных алгоритмов
$\varphi\text{-}\sigma\|_{f}^{fs}$	сжатие (f и fs — файл(ы) до и после сжатия соответственно)	$\varphi\text{-}\beta_1\|_{Ms}^{M}$	постобработка
$\varphi\text{-}\alpha_1\|_{M}^{Ms}$	предварительная обработка	$\varphi\text{-}\beta_2\|_{Ms}^{M}$	обратный алгоритм универсальной обработки
$\varphi\text{-}\alpha_2\|_{M}^{Ms}$	универсальная обработка (после или вместо предварительной)	$\varphi\text{-}\beta_3\|_{Ms}^{M}$	одиночный метод декомпрессии
$\varphi\text{-}\alpha_3\|_{M}^{Ms}$	одиночный метод сжатия	$\varphi\text{-}\beta_4\|_{Ms}^{M}$	последовательность методов декомпрессии
$\varphi\text{-}\alpha_4\|_{M}^{Ms}$	последовательность методов сжатия	$\varphi\text{-}\beta_5\|_{M}^{Md,Mf,s}$	подготовка к декодированию
$\varphi\text{-}\alpha_5\|_{M}^{Md,Mf,s}$	подготовка к кодированию	$\varphi\text{-}\beta_6\|_{Mb}^{Mc}$	декодирование
$\varphi\text{-}\alpha_6\|_{Mc}^{Mb}$	кодирование	$\varphi\text{-}\beta_7\|_{Ms}^{M}$	декомпрессия методом арифметического кодирования
$\varphi\text{-}\alpha_7\|_{M}^{Ms}$	арифметическое кодирование		

• в правилах вывода с аксиомой в левой части $\sigma \in U_1$ отношений подстановки принимаются $\vec{p} = \langle 0.5, 0.2, 0.5, 0.3, 0.05, 0.1, 0.2, 0.25, 0.4, 0.5\rangle$, $maxn = 50$;

• реализация КПС двойственного конструктора алгоритмов сжатия представляет собой сконструированную пару алгоритмов сжатия и декомпрессии.

Рассмотим правила вывода аксиоматики Λ_5.

Отношения подстановок $\bar{s}_1$ и $\tilde{s}_1$ предназначены для формирования алгоритмов сжатия и декомпрессии соответственно. Отметим, что алгоритм декомпрессии формируется одновременно с алгоритмом сжатия в прямом порядке, а выполняется — в обратном. До выполнения подстановки доступность всех правил, кроме пятого $\tilde{s}_5$, устанавливается в единицу, после чего общее количество использованных базовых алгоритмов устанавливается в ноль, и обнуляется вектор количества использованных базовых алгоритмов

$$\bar{s}_1 = \langle {}_{maxn,\vec{q},\vec{p},m}\sigma|_{f\ \vec{d}}^{fs} \to B_{17}^{0}|_{f}^{M} \cdot \alpha_1|_{M}^{Mo} \cdot \alpha_2|_{Mo}^{Mo_1} \cdot \alpha_3|_{Mo_1}^{Ms} \cdot B_9^0|_{Ms}^{fs}\rangle,$$

$$\tilde{s}_1 = \langle {}_{maxn,\vec{q},\vec{p},m}\sigma|_{f\ \vec{d}}^{fs} \to (B_{17}^{-1})^0|_{M}^{f} \cdot \beta_1|_{Mo}^{M} \cdot \beta_2|_{Mo_1}^{Mo} \cdot \beta_3|_{Ms}^{Mo_1} \cdot (B_9^{-1})^0|_{fs}^{Ms}\rangle,$$

$$g_1 = \langle \tau_0 g_{1,1}, \tau_1 g_{1,2}\rangle,$$

$$\tau_0 g_{1,1} = \langle d_i := 1\ \forall i \in [1;9];\ d_5 := 0\rangle,\quad \tau_1 g_{1,2} = \langle m := 0,\ q_i := 0\ \forall i \in [1;17]\rangle.$$

Операция подстановки с операндом $\bar{s}_2$ формирует алгоритм предварительной обработки. В $\tau_1 g_{2,2}$ задан набор операций, которые выполнятся после подстановки: многоместная операция выбора $\otimes$ номера алгоритма, инкремента количества алгоритмов B_1^0 или B_3^0 в конструкции формируемого алгоритма сжатия и общего количества алгоритмов

$$\bar{s}_2 = \langle {}_{maxn,\vec{q},\vec{p},m}\alpha_1|_{M\ \vec{d}}^{M_0} \to (A_2^0|_{p,(0,p_1)}^{:p_1 B_1^0|_{M}^{Mo}} : p_1 B_1^0|_{M}^{Mo}) \cdot (A_2^0|_{p,(p_1,1)}^{:p_3 B_3^0|_{M}^{M}} : p_3 B_3^0|_{M}^{M})\rangle,$$

$$\tilde{s}_2 = \langle {}_{maxn,\vec{q},\vec{p},m}\beta_1|_{M_0\ \vec{d}}^{M} \to (A_2^0|_{p,(0,p_1)}^{:p_1 (B_1^{-1})^0|_{Mo}^{M}} : p_1 (B_1^{-1})^0|_{Mo}^{M}) \cdot$$

$$\cdot (A_2^0|_{p,(p_1,1)}^{:p_3 (B_3^{-1})^0|_{M}^{M}} : p_3 (B_3^{-1})^0|_{M}^{M})\rangle,$$

$$g_2 = \langle \tau_0 g_{2,1}, \tau_1 g_{2,2}\rangle,\quad \tau_0 g_{2,1} = \langle p := \vee(0,1)\rangle,$$

$$\tau_1 g_{2,2} = \langle i := \otimes(2, p, (1;3), ((0, p_1); (p_1, 1)));\quad m := m+1;\ q_i := q_i + 1\rangle.$$

Отношение подстановки $\bar{s}_3$ определяет выбор алгоритма универсальной обработки на основании вероятности p:

$$\bar{s}_3 = \langle {}_{maxn,\vec{q},\vec{p},m}\alpha_2|{}^{M_0}_{M\ \vec{d}} \to (A_2^0|^{:}_{p,(0,p_2)} : {}_{p_2}B_2^0|^{Mo}_{M}) \cdot$$

$$\cdot (A_2^0|^{:}_{p,(p_2,p_4)} : {}_{p_4}B_4^0|^{Mo}_{M}) \cdot (A_2^0|^{:}_{p,(p_4,1)} : {}_{p_3}B_3^0|^{M}_{M})\rangle,$$

$$\widetilde{s}_3 = \langle {}_{maxn,\vec{q},\vec{p},m}\beta_2|^{M}_{M_0\ \vec{d}} \to (A_2^0|^{:}_{p,(0,p_2)} : {}_{p_2}(B_2^{-1})^0|^{M}_{Mo}) \cdot$$

$$\cdot (A_2^0|^{:}_{p,(p_2,p_4)} : {}_{p_4}(B_4^{-1})^0|^{M}_{Mo}) \cdot (A_2^0|^{:}_{p,(p_4,1)} : {}_{p_3}(B_3^{-1})^0|^{M}_{M})\rangle,$$

$$g_3 = \langle \tau_0 g_{3,1}, \tau_1 g_{3,2}\rangle, \quad \tau_0 g_{3,1} = \langle p := \vee(0,1)\rangle,$$

$$\tau_1 g_{3,2} = \langle i := \otimes(3,\ p,\ (2;4;3),((0,\ p_2);(p_2,\ p_4);(p_4,1));\ m := m+1;\ q_i := q_i + 1\rangle.$$

Для формирования алгоритма сжатия задана косвенная рекурсия, которая завершается либо случайным образом, либо по достижении максимально возможного количества базовых алгоритмов в форме

$$\bar{s}_4 = \langle {}_{maxn,\vec{q},\vec{p},m}\alpha_3|^{Ms}_{M\ \vec{d}} \to (A_2^0|^{:\,{}_{p_5}B_5^0|^{Mo}_{M}}_{p,(0,p_5)} : {}_{p_5}B_5^0|^{Mo}_{M}) \cdot (A_2^0|^{:\,{}_{p_6}B_6^0|^{Mo}_{M}}_{p,(p_5,p_6)} : {}_{p_6}B_6^0|^{Mo}_{M}) \cdot$$

$$\cdot (A_2^0|^{:}_{p,(p_6,p_7)} : {}_{p_7}B_7^0|^{Ms}_{M}) \cdot (A_2^0|^{:\,{}_{p_8}B_8^0|^{Ms}_{M}}_{p,(p_7,p_8)} : {}_{p_8}B_8^0|^{Ms}_{M}) \cdot (A_2^0|^{:\,{}_{p_9}\alpha_8|^{Ms}_{M}}_{p,(p_8,1)} : {}_{p_9}\alpha_7|^{Ms}_{M}) \cdot \alpha_4|^{fs}_{Ms}\rangle,$$

$$\widetilde{s}_4 = \langle {}_{maxn,\vec{q},\vec{p},m}\beta_3|^{M}_{Ms\ \vec{d}} \to (A_2^0|^{:\,{}_{p_5}(B_5^{-1})^0|^{M}_{Mo}}_{p,(0,p_5)} : {}_{p_5}(B_5^{-1})^0|^{M}_{Mo}) \cdot$$

$$\cdot (A_2^0|^{:\,{}_{p_6}(B_6^{-1})^0|^{M}_{Mo}}_{p,(p_5,p_6)} : {}_{p_6}(B_6^{-1})^0|^{M}_{Mo}) \cdot (A_2^0|^{:\,{}_{p_7}(B_7^{-1})^0|^{M}_{Ms}}_{p,(p_6,p_7)} : {}_{p_7}(B_7^{-1})^0|^{M}_{Ms}) \cdot$$

$$\cdot (A_2^0|^{:\,{}_{p_8}(B_8^{-1})|^{M}_{Ms}}_{p,(p_7,p_8)} : {}_{p_8}(B_8^{-1})|^{M}_{Ms}) \cdot (A_2^0|^{:\,{}_{p_9}\beta_8|^{M}_{Ms}}_{p,(p_8,1)} : {}_{p_9}\beta_7|^{M}_{Ms}) \cdot \beta_4|^{Ms}_{fs}\rangle,$$

$$g_4 = \langle \tau_0 g_{4,1}, \tau_1 g_{4,2}\rangle, \quad \tau_0 g_{4.1} = \langle \div(m = maxn+1, 2, (d_4 := 0, d_5 := 1)),\ p := \vee(0,1)\rangle,$$

$$\tau_1 g_{4,2} = \langle i := \otimes(5,\ p,\ (5;6;7;8;0),((0,p_5);(p_5,p_6);(p_6,p_7);(p_7,p_8);(p_8,1)));$$

$$m := m+1;\ \div(i \neq 0,1,\ q_i := q_i + 1),\ M := Ms\rangle.$$

Если достигнуто максимальное количество базовых алгоритмов, входящих в конструкцию, то необходимо прекратить их подстановку

$$\bar{s}_5 = \langle {}_{maxn,\ \vec{q},\ \vec{p},\ m}\alpha_3|^{Ms}_{M\ \vec{d}} \to \varepsilon\rangle, \quad \widetilde{s}_5 = \langle {}_{maxn,\ \vec{q},\ \vec{p},\ m}\beta_3|^{Ms}_{M\ \vec{d}} \to \varepsilon\rangle, \quad \tau_1 g_5 = \langle d_5 := 0, d_4 := 1\rangle.$$

Следующее отношение позволяет случайным образом выбрать, завершить или продолжать рекурсию формирования адаптивного алгоритма сжатия:

$$\bar{s}_6 = \langle {}_{maxn,\vec{q},\vec{p},m}\alpha_4|^{Ms}_{M\ \vec{d}} \to (A_2^0|^{:\,{}_{p_{10}}\alpha_3|^{Mo}_{M}}_{p,(0,p_{10})} : {}_{p_{10}}\alpha_3|^{Mo}_{M})\rangle,$$

$$\widetilde{s}_6 = \langle {}_{maxn\,\vec{q},\vec{p},m}\beta_4|^{M}_{Ms\,\vec{d}} \to (A_2^0|^{:\,{}_{p_{10}}(\beta_3)^0|^{M}_{Mo})}_{p,(0,p_{10})} : {}_{p_{10}}(\beta_3)^0|^{M}_{Mo}\rangle\rangle, \quad \tau_0 g_6 = \langle p := \vee(0,1)\rangle.$$

Отношение для подстановки базового алгоритма кодирования массива имеет вид

$$\bar{s}_7 = \langle {}_{maxn,\vec{q},\vec{p},m}\alpha_7|^{Ms}_{M\ \vec{d}} \to \alpha_5|^{Md,Mf,s}_{M} \cdot B_{10}^0|^{Mc}_{M,Md,Mf,s} \cdot \alpha_6|^{Mb}_{Mc}\rangle,$$

$$\widetilde{s}_7 = \langle {}_{maxn,\vec{q},\vec{p},m}\beta_7|^{Ms}_{M\ \vec{d}} \to \beta_5|^{M}_{Md,Mf,s} \cdot (B_{10}^{-1})^0|^{M,Md,Mf,s}_{Mc} \cdot \beta_6|^{Mc}_{Mb}\rangle,$$

$$\tau_1 g_7 = \langle q_{10} := q_{10} + 1\rangle.$$

Вместо абстрактного оператора кодирования массивов последовательно подставляются базовые алгоритмы обработки символьного массива для его кодирования, следовательно, увеличивается на единицу счетчик количества вхождений данных алгоритмов в форму

$$\bar{s}_8 = \langle {}_{maxn,\vec{q},\vec{p},m}\alpha_5 |^{Md,Mf,s}_{M\ \vec{d}} \to B^0_{11}|^{Ma,Mb}_{Ma,Mb} \cdot B^0_{12}|^{Ma}_{M} \cdot B^0_{13}|^{Mb,Md,Mf}_{Ma} \cdot B^0_{14}|^{Ma,Mb}_{Ma,Mb} \rangle,$$

$$\tilde{s}_8 = \langle {}_{maxn,\vec{q},\vec{p},m}\beta_5 |^{M}_{Md,Mf,s\ \vec{d}} \to (B^{-1}_{11})^0|^{Ma,Mb}_{Ma,Mb} \cdot (B^{-1}_{12})^0|^{M}_{Ma} \cdot$$

$$\cdot (B^{-1}_{13})^0|^{Ma}_{Mb,Md,Mf} \cdot (B^{-1}_{14})^0|^{Ma,Mb}_{Ma,Mb} \rangle,$$

$$\tau_1 g_8 = \langle q_i := q_i + 1\ \forall i = [11;14] \rangle.$$

Абстрактный оператор заменяется последовательностью базовых алгоритмов $B^0_{15}|^{Mc}_{M,Md,Mf,s}$ и $B^0_{16}|^{Mc}_{M,Md,Mf,s}$, следовательно, увеличивается число включений данных алгоритмов в форму

$$\bar{s}_9 = \langle {}_{maxn,\vec{q},\vec{p},m}\alpha_6 |^{Mb}_{Mc\ \vec{d}} \to B^0_{15}|^{Mc}_{M,Md,Mf,s} \cdot B^0_{16}|^{Mc}_{M,Md,Mf,s} \rangle,$$

$$\tilde{s}_9 = \langle {}_{maxn,\vec{q},\vec{p},m}\beta_6 |^{Mc}_{Mb\ \vec{d}} \to (B^{-1}_{15})^0|^{M,Md,Mf,s}_{Mc} \cdot (B^{-1}_{16})^0|^{M,Md,Mf,s}_{Mc} \rangle,$$

$$\tau_1 g_9 = \langle q_i := q_i + 1\ \ \forall i = [15,16] \rangle.$$

Реализация двойственного конструктора существенно зависит от вектора рекомендуемых вероятностей, влияющих на выбор алгоритмов предварительной обработки и сжатия. Рассмотрим примеры реализации двойственного конструктора.

Пример 1. Сконструированный алгоритм сжатия $B^0_{17}|^{M}_{f} \cdot B^0_{1}|^{Ms}_{M} \cdot B^0_{4}|^{Ms}_{M} \cdot B^0_{7}|^{Ms}_{M} \cdot B^0_{5}|^{Ms}_{M} \cdot B^0_{9}|^{fs}_{M,fs}$ является конкатенацией соответственно базовых алгоритмов чтения данных из файла, предобработки arcDelta2 и RLE, сжатия методами arcLZMA и arcDeflate, а также сохранения сжатых данных в новый файл. Алгоритм декомпрессии состоит из соответствующих обратных алгоритмов:

$$(B^{-1}_{17})^0|^{f}_{M} \cdot (B^{-1}_{1})^0|^{M}_{Ms} \cdot (B^{-1}_{4})^0|^{M}_{Ms} \cdot (B^{-1}_{7})^0|^{M}_{Ms} \cdot (B^{-1}_{5})^0|^{M}_{Ms} \cdot (B^{-1}_{9})^0|^{M,fs}_{fs}.$$

Пример 2. Сконструированный алгоритм сжатия $B^0_{17}|^{M}_{f} \cdot B^0_{2}|^{M}_{M} \cdot B^0_{4}|^{Ms}_{M} \cdot B^0_{7}|^{Ms}_{M} \cdot B^0_{5}|^{Ms}_{M} \cdot B^0_{8}|^{Ms}_{M} \cdot B^0_{5}|^{Ms}_{M} \cdot B^0_{6}|^{Ms}_{M} \cdot B^0_{9}|^{fs}_{M,fs}$ является конкатенацией соответственно базовых алгоритмов чтения данных из файла, предобработки arcBase64 и RLE, сжатия методами arcLZMA, arcDeflate, GZip, arcLZMA, arcBZip и сохранения сжатых данных в новый файл. Алгоритм декомпрессии состоит из соответствующих обратных алгоритмов:

$$(B^{-1}_{17})^0|^{f}_{M} \cdot (B^{-1}_{2})^0|^{M}_{M} \cdot (B^{-1}_{4})^0|^{M}_{Ms} \cdot (B^{-1}_{7})^0|^{M}_{Ms} \cdot (B^{-1}_{5})^0|^{M}_{Ms} \cdot (B^{-1}_{8})^0|^{M}_{Ms} \cdot$$

$$\cdot (B^{-1}_{5})^0|^{M}_{Ms} \cdot (B^{-1}_{6})^0|^{M}_{Ms} \cdot (B^{-1}_{9})^0|^{M,fs}_{fs}.$$

ПРЕОБРАЗОВАТЕЛЬ АЛГОРИТМОВ СЖАТИЯ

Преобразователь формирует и выполняет конструктивный процесс сжатия согласно сконструированного конструктором алгоритма. Внутренний исполнитель преобразователя одновременно разбирает конструкцию алгоритма сжатия, формирует и выполняет процесс сжатия. Фактически преобразователь содержит

программно-аппаратный комплекс, с помощью которого осуществляется проверка алгоритма, построение исполнимого файла, запуск и выполнение полученной программы. Параметры программно-аппаратного комплекса задаются аксиоматикой в качестве атрибутов.

Рассмотим представление преобразователя средствами КПС.

Специализация ОКПС для представления преобразователя алгоритмов сжатия имеет вид

$$C = \langle M, \Sigma, \Lambda \rangle_S \mapsto C_{PAC} = \langle M_{PAC}, \Sigma_{PAC}, \Lambda_{PAC} \rangle,$$

где $\Lambda_{PAC} = \Lambda \bigcup \Lambda_6 \bigcup \Lambda_7$, $\Lambda_6 = \{M_{PAC} \supset T_2 \bigcup N_2$, $\Sigma_{PAC} = \Sigma \bigcup \Sigma_{PAC_1}$, $\Sigma_{PAC_1} = \{\Xi, \Theta, \Phi\}$, $\Xi = \{\cdot\}$, $\Theta = \{\Rightarrow, |\Rightarrow, ||\Rightarrow\}$, $\Phi = \{:=, =, +, \neq, \div, \uparrow\}\}$.

Частичная аксиоматика Λ_7 содержит следующие дополнения:

• терминальный алфавит, включающий терминалы конструктора алгоритмов сжатия (для разбора конструкций) и терминалы со значениями соответствующих процессов выполнения базовых алгоритмов $T_2 = \{O = \{B_i^0|_X^Y\} \bigcup \{{}_{\Re}\widetilde{B}_i^0|_X^Y\}\}$, где $\Re$ — атрибуты внешней среды выполнения процесса, а также нетерминальный алфавит с символами для алгоритма сжатия и процесса выполнения соответствующего алгоритма $N_2 = \{{}_{As}\omega, {}_{As}\widetilde{\omega}\}$, где As — конструкция алгоритма сжатия;

• правила подстановки $\vec{d}\, p_i : \langle s_i, g_i \rangle \in \Psi_P$, где $\Psi_P \subset \Lambda_{PAC}$, s_i — отношения подстановки для разбора конструкции алгоритма сжатия и преобразования базовых алгоритмов в процесс их выполнения, g_i — операции над атрибутами, $\vec{d}$ — вектор доступности правил;

• результатом операции $(\uparrow(As, i))$ выбора i-го элемента конструкции $As = \prod_i B_i^0$ является базовый алгоритм B_i, где As — адаптивный алгоритм сжатия;

• операция $\div$ определяется как в C_{KAC}.

Для выполнения интерпретации структуры C_{PAC} воспользуемся представленной далее БАС

$$C_{A,PAC} = \langle M_{A,PAC}, V_{A,PAC}, \Sigma_{A,PAC}, \Lambda_{A,PAC} \rangle,$$

где $M_{A,PAC} = M'_{A,KAC} \bigcup M_{A,KAC}$ — носитель, $\Sigma_{A,PAC} = \Sigma_{A,KAC}$ — сигнатура и $\Lambda_{A,PAC}$ — аксиоматика.

Алгоритмическая структура $C_{A,PAC}$ содержит следующие алгоритмы:

• $A_1^0|_{A_i,A_j}^{A_i \cdot A_j}$, $A_3|_{l_h,l_q,f_i}^{f_j}$, $A_4|_{f_i,\Psi}^{f_j}$, $A_5|_{\sigma,\Psi}^{\overline{\Omega}}$, $A_6^0|_a^a$, $A_7^0|_a^b$, $A_{10}^0|_{a,b}^c$, $A_{11}^0|_{a,b}^c$, $A_{12}|_{c,n,L}^L$ такие же, как в $C_{A,KAC}$;

• $A_8^0|_{As,i}^{\hat{B}|_X^Y}$ — операция выбора i-го элемента конструкции As.

Рассмотим интерпретацию формальной структуры преобразователя алгоритмов сжатия

$$C_{PAC} = \langle M_{PAC}, \Sigma_{PAC}, \Lambda_{PAC} \rangle_I \mapsto_{I, C_{A,PAC}} C_{I,PAC} = \langle M_{PAC}, \Sigma_{PAC}, \Lambda_{I,PAC} \rangle,$$

где $\Lambda_{I,PAC} = \Lambda_{PAC} \bigcup \Lambda_8$, $\Lambda_8 = \{(A_1^0|_{A_i,A_j}^{A_i \cdot A_j} \hookleftarrow \cdot), (A_3|_{l_h,l_q,f_i}^{f_j} \hookleftarrow \Rightarrow), (A_4|_{f_i,\Psi}^{f_j} \hookleftarrow |\Rightarrow), (A_5|_{\sigma,\Psi}^{\overline{\Omega}} \hookleftarrow ||\Rightarrow), (A_6^0|_a^a \hookleftarrow +), (A_7^0|_a^b \hookleftarrow :=), (A_8^0|_{As,i}^{\hat{B}|_X^Y} \hookleftarrow \uparrow), (A_{10}^0|_{a,b}^c \hookleftarrow \neq), (A_{11}^0|_{a,b}^c \hookleftarrow =), (A_{12}^0|_{c,n,L}^L \hookleftarrow \div)$.

Конкретизируем представленную структуру преобразователя алгоритмов сжатия

$$ {}_{I}C_{PAC} = \langle M_{PAC}, \Sigma_{PAC}, \Lambda_{I,PAC} \rangle_{K} \mapsto C_{K,PAC} = $$
$$ = \langle M_{PAC}, \Sigma_{PAC}, \Lambda_{I,PAC} \cup \Lambda_9 \cup \Lambda_{10} \rangle, $$

$$ \Lambda_9 = \{T_2 = O \cup \{\widetilde{B}_1^0|_M^{M_0},\ \widetilde{B}_2^0|_M^{M_0}, \widetilde{B}_3^0|_M^{M}, \widetilde{B}_4^0|_M^{M_0}, \{\widetilde{B}_{\mathrm{i}}^0|_M^{Ms}\ \forall i = \overline{5,8}\}; $$

$$ \widetilde{B}_9^0|_M^{fs},\ \widetilde{B}_{10}^0|_{M,Md,Mf,s}^{Mc}, \widetilde{B}_{11}^0|_{Ma,Mb}^{Ma,Mb}, \widetilde{B}_{12}^0|_M^{Ma}, \widetilde{B}_{13}^0|_{Ma}^{Mb,Af,Bf}, \widetilde{B}_{14}^0|_{Ma,Mb}^{Ma,Mb}, $$

$$ \widetilde{B}_{15}^0|_{Ma}^{Mc}, \widetilde{B}_{16}^0|_{Ma}^{Mb}, \widetilde{B}_{17}^0|_f^M\ \},\ N_2 = \{\omega, \tilde{\omega}\}, U = \{\omega\}, \Psi_P\}, $$

где ${}_{\Re}\widetilde{B}_i^0|_X^Y$ — процесс выполнения базового алгоритма $B_i^0|_X^Y \in T_1$.

Частичная аксиоматика Λ_{10} содержит следующие дополнения:

• в правилах вывода с аксиомой в левой части принимается $j = 0$;

• для преобразования и выполнения алгоритмов сжатия используется компьютер с процессором Intel, операционной системой Windows и жестким диском с достаточным свободным пространством (атрибутика внешней среды);

• правила вывода представляются в виде

$$ \Psi_P = \{s_0 = \langle {}_{As,j}\omega\ \vec{d} \to \varepsilon, {}_{As,j}\ \tilde{\omega}\ \vec{d} \to \varepsilon\rangle, $$
$$ \langle s_i = \langle {}_{As,j}\omega\ \vec{d} \to B_i^0|_X^Y \cdot {}_{As,j}\omega,\ {}_{As,j}\tilde{\omega}\ \vec{d} \to {}_{\Re}\widetilde{B}_i^0|_X^Y \cdot {}_{As,j}\tilde{\omega}\rangle, $$

$$ {}_{t_0}g_i = \langle \hat{B}|_X^Y := \uparrow(As, j), \div(\hat{B}|_X^Y = B^0|_X^Y, 1, d_i := 1), \div(\hat{B}|_X^Y \neq B^0|_X^Y, 1, d_i := 0\rangle, $$
$$ {}_{\tau_1}g_i = \langle j := j+1\rangle\ \ \forall i = 1, \ldots, 17\rangle, $$

где $\hat{B}|_Y^X \in M_{PAC}$ — текущий базовый алгоритм конструкции.

На этапе конкретизации преобразователя внешний исполнитель выполняет согласование с конструктором алгоритмов сжатия по терминальному алфавиту (подмножество базовых алгоритмов конструктора включается в терминальный алфавит преобразователя).

Реализация преобразователя алгоритмов сжатия представляет собой множество процессов выполнения алгоритмов сжатия. Например, конструктивный процесс представленного в примере 1 алгоритма сжатия является конкатенацией базовых процессов выполнения алгоритмов: $\widetilde{B}_{17}^0|_f^M \cdot \widetilde{B}_1^0|_M^{Ms} \cdot \widetilde{B}_3^0|_M^M \cdot \widetilde{B}_7^0|_M^{Ms} \cdot \widetilde{B}_5^0|_M^{Ms} \cdot \widetilde{B}_9^0|_{M,fs}^{fs}$.

АДАПТЕР АЛГОРИТМОВ СЖАТИЯ

Структурная адаптация алгоритмов сжатия включает [14]: их синтез с помощью двойственного конструктора алгоритмов сжатия, выполнение синтезированных алгоритмов с измерениями степени сжатия, а также анализ и выбор наилучшего алгоритма сжатия.

Синтез заключатся в подготовке к адаптации и в конструировании алгоритмов. На этом этапе конструктор формирует алгоритмы сжатия, используя информацию, подготовленную адаптером, который в данном случае является внешним исполнителем по отношению к конструктору.

Далее алгоритмы сжатия поступают на вход преобразователя (исполнителя), который при необходимости транслирует (интерпретирует) их и выполняет на конкретных наборах входных данных. Адаптер содержит средства для измерения показателей эффективности сформированных алгоритмов сжатия — степени сжатия, а также операцию для определения наилучшего алгоритма сжатия из сконструированных.

Процесс адаптации осуществляется посредством перерасчета рекомендуемых вероятностей использования базовых алгоритмов сжатия. Первоначально они одинаковы. По результатам анализа рекомендуемая вероятность более эффективных алгоритмов повышается, а остальных — понижается. Методика определения рекомендуемых вероятностей описана в [15]. После того, как сконструировано заданное количество алгоритмов сжатия, они выполняются и для каждого из них измеряется показатель эффективности. Затем выполняется перерасчет рекомендуемых вероятностей, определяются наилучший алгоритм и необходимость продолжения адаптации.

Специализация адаптера алгоритмов сжатия. Рассмотрим специализацию ОКПС для представления адаптера алгоритмов сжатия

$$C = \langle M, \Sigma, \Lambda \rangle_S \mapsto C_{AAC} = \langle M_{AAC}, \Sigma_{AAC}, \Lambda_{AAC} \rangle,$$

где $\Lambda_{AAC} = \Lambda \bigcup \Lambda_{11} \bigcup \Lambda_{12}$, $\Lambda_{11} = \{M_{AAC} \supset T_3 \bigcup N_3$, $\Sigma_{AAC} = \Sigma_{KAC} \bigcup \Sigma_{AAC_1}$, $\Sigma_{AAC_1} = \{\Xi, \Theta, \Phi\}$, $\Xi = \{\cdot, :\}$, $\Phi = \{+, :=, -, \Diamond, \nabla, \otimes, =, >, /, \div, \Updownarrow, \hbar\}$, $\Theta = \{\overset{*}{\Rightarrow}, |\overset{*}{\Rightarrow}, |\overset{*}{\models}\}\}$. Здесь сигнатура операций определения: $\Diamond$ — лучшего алгоритма сжатия, ∇ — степени сжатия файла(ов), $\Updownarrow$ — размера файла, / — остатка от деления, $\hbar$ — элемента по его номеру в списке, +, ÷, :=, = такие, как в $C_{I,C_{A,KAC}} C_{KAC}$.

Частичная аксиоматика Λ_{12} содержит следующие дополнения:

- носитель M_{AAC} имеет сформированные конструктором алгоритмы сжатия $As_k = \left(\prod_i \{B_i^0 |_{X_i}^{Y_i}\} \right)_k$ и декомпрессии $Ds_k = \left(\prod_i \{(B^{-1})_i^0 |_{Y_i}^{X_i}\} \right)_k$, в конструкциях алгоритмов сжатия содержатся атрибуты: образ алгоритма $(\vec{q} \hookleftarrow As_k) = [Q_{i_1}, Q_{i_2}, \ldots, Q_{i_k}]_k$ — вектор количества включений в k-й алгоритм сжатия i-го базового алгоритма, и степень сжатия данных указанным алгоритмом $s \hookleftarrow As_k$;
- носитель M_{AAC} включает сформированные адаптером списки алгоритмов сжатия ${}_hR$ и декомпрессии ${}_hRd$, где $h \in N$ — количество элементов в списке;
- терминальный алфавит содержит множество алгоритмов, необходимых для процесса адаптации $T_3 = \{B_i |_{X_i}^{Y_i}\}$, а нетерминальный — символы для обозначения этапов процесса адаптации;
- двухместная операция частичного вывода $l^* = (|\overset{*}{\Rightarrow}(\Psi, l))$ $(|\overset{*}{\Rightarrow} \in \Theta)$ выполняется так же, как в двойственном конструкторе;
- завершает процесс адаптации операция полного вывода, условием окончания которого является отсутствие нетерминалов в форме.

Интерпретация адаптера алгоритмов сжатия. Для определения интерпретации структуры C_{AAC} воспользуемся БАС

$$C_{A,AAC} = \langle M_{A,AAC}, V_{A,AAC}, \Sigma_{A,AAC}, \Lambda_{A,AAC} \rangle,$$

где $M_{A,AAC} = M'_{A,AAC} \bigcup M_{A,KAC} \bigcup M_{A,PAC}$ и $M_{A,AAC}$ — носитель, $\Sigma_{A,AAC} = \Sigma_{A,AAC} \bigcup \Sigma_{A,KAC} \bigcup \Sigma_{A,PAC}$ — сигнатура и $\Lambda_{A,AAC} = \Lambda'_{A,AAC} \bigcup \Lambda_{A,KAC} \bigcup \bigcup \Lambda_{A,PAC}$ — аксиоматика. Алгоритмическая структура $C_{A,AAC}$ содержит следующие алгоритмы:

- $A_1^0 |_{A_i, A_j}^{A_i \cdot A_j}$, $A_6^0 |_a^a$, $A_7^0 |_a^b$, $A_9 |_{p,k,\overline{L},\overline{L}}^{c}$, $A_{11}^0 |_{a,b}^c$, $A_{12} |_{c,n,L}^{L}$ такие, как в $C_{A,KAC}$;
- $A_{13} |_{l_h, l_q, f_i}^{f_j}$, $A_{14} |_{f_i, \Psi_A}^{f_j}$, $A_{15} |_{\sigma,\Psi}^{\overline{\Omega}}$ такие, как $A_3 |_{l_h, l_q, f_i}^{f_j}$, $A_4 |_{f_i,\Psi}^{f_j}$, $A_5 |_{\sigma,\Psi}^{\overline{\Omega}}$ соответственно в $C_{A,KAC}$ с учетом дополнений аксиоматики Λ_{11} и Λ_{12};

• $A_{16}^{0}|_{a,b}^{c}$ — определения остатка от деления a на b;

• $A_{17}^{0}|_{a,b}^{c}$ — сравнения, где c принимает значение *true*, если $a > b$, и *false* в противном случае;

• $A_{18}^{0}|_{f}^{u}$ — измерения размера файла f, где u — размер файла;

• $A_{19}^{0}|_{a,b}^{c}$ — определения частного деления b на a;

• $A_{20}|_{f,fs}^{s}$ — расчета степени сжатия (алгоритм является составным):

$$A_{20}|_{f,fs}^{s} = A_{18}^{0}|_{f}^{s_m} \cdot A_{18}^{0}|_{fs}^{s_{ms}} \cdot A_{19}^{0}|_{s_{ms},s_m}^{s};$$

• $A_{21}|_{s\hat{R}_1}^{best}$ — определения номера наилучшего по показателю эффективности алгоритма сжатия;

• $A_{22}^{0}|_{R,best}^{Ad}$ — выделения алгоритма сжатия (декомпрессии) Ad из списка R по номеру $best$.

Используя БАС $C_{A,AAC}$, выполняем интерпретацию формальной структуры адаптера алгоритмов сжатия:

$$C_{AAC} = \langle M_{AAC}, \Sigma_{AAC}, \Lambda_{AAC} \rangle_{I} \mapsto {}_{I,C_{A,AAC}} C_{I,AAC} = \langle M_{AAC}, \Sigma_{AAC}, \Lambda_{AAC} \cup \Lambda_{13} \rangle,$$

где $\Lambda_{13} = \{(A_1^0|_{A_i,A_j}^{A_i \cdot A_j} \hookleftarrow \cdot),\ (A_6^0|_{a}^{a} \hookleftarrow +),\ (A_7^0|_{a}^{b} \hookleftarrow :=),\ (A_9|_{p,k,\bar{L},\bar{=}}^{c} \hookleftarrow \otimes),\ (A_{11}^0|_{a,b}^{c} \hookleftarrow =),$ $(A_{12}|_{c,n,L}^{L} \hookleftarrow \div),\ (A_{13}|_{l_h,l_q,f_i}^{f_j} \hookleftarrow \overset{*}{\Rightarrow}),\ (A_{14}|_{f_i,\Psi}^{f_j} \hookleftarrow |\overset{*}{\Rightarrow}),\ (A_{15}|_{\sigma,\Psi}^{\overline{\Omega}} \hookleftarrow ||\overset{*}{\Rightarrow}),\ (A_{16}^0|_{a,b}^{c} \hookleftarrow /),$ $(A_{17}^0|_{a,b}^{c} \hookleftarrow >),\ (A_{18}^0|_{f}^{u} \hookleftarrow \Updownarrow),\ (A_{19}^0|_{a,b}^{c} \hookleftarrow :),\ (A_{20}|_{f,fs}^{s} \hookleftarrow \nabla),\ (A_{21}|_{s\hat{R}}^{best} \hookleftarrow \Diamond),$ $(A_{22}^0|_{R,best}^{Ad} \hookleftarrow \hbar)$.

Конкретизация адаптера алгоритмов сжатия. Рассмотрим конкретизацию адаптера алгоритмов сжатия для его реализации, представленной в [7]

$${}_{I}C_{AAC} = \langle M_{AAC}, \Sigma_{AAC}, \Lambda_{I,AAC} \rangle_{K} \mapsto C_{K,AAC} = \langle M_{AAC}, \Sigma_{AAC}, \Lambda_{K,AAC} \rangle,$$

где $\Lambda_{K,AAC} = \Lambda_{AAC} \cup \Lambda_{I,KAC} \cup \Lambda_{13} \cup \Lambda_{14}$, $\Lambda_{13} = \{T_3 = \{B_{18}^0|_{R,i}^{As_i},\ B_{19}^0|_{f,\vec{p}}^{As|_f^{fs},Ds|_{fs}^{f}},$ $B_{20}^0|_{X}^{R}, B_{21}^0|_{As|_f^{fs},f}^{fs},\ B_{22}^0|_{R}^{s\hat{R}},\ B_{23}^0|_{s\hat{R},h}^{s\hat{R}_1},\ B_{24}^0|_{s\hat{R}_1}^{K},\ B_{25}^0|_{K}^{r},\ B_{26}^0|_{K,r}^{\vec{s}_{\max}},\ B_{27}^0|_{K}^{s_{\min}},$ $B_{28}^0|_{K,\vec{s}_{\max},s_{\min}}^{\vec{l}},\ B_{29}^0|_{K,\vec{l}}^{\overrightarrow{Pq}},\ B_{30}^0|_{\overrightarrow{Pq}}^{\overrightarrow{Pk}},\ B_{31}^0|_{K,r,\overrightarrow{Pk},\vec{l}}^{\vec{p}},\ B_{32}^0|_{Ad,Dd,\vec{p}}^{rez,Ad,Dd,\vec{p}}\}$, $N_3 =$ $= \{\sigma, \chi, \delta, \vartheta, \xi, \lambda\}$, $U_3 = \{\sigma\}$, $\Psi_A = \{s_i, g_i\}\}$. Здесь N_3 включает нетерминалы — абстрактные алгоритмы, а T_3 — процессы выполнения алгоритмов $B_j^0|_{X_j}^{Y_j}$:

• $B_{18}^0|_{{}_hR,i}^{As_i|_f^{fs}}$ — извлечение алгоритма $As_i|_f^{fs}$ из списка ${}_hR$, где ${}_hR = \langle As_i|_f^{fs} \rangle$ — список алгоритмов сжатия, сформированных двойственным конструктором $As_i|_f^{fs} \in \Omega(C_{KAC})$, атрибут h — количество алгоритмов в списке;

• $B_{19}^0|_{f,\vec{p}}^{As|_f^{fs},Ds|_{fs}^{f}}$ — формирование алгоритма сжатия и декомпрессии (двойственным конструктором), где f — файл(ы) для сжатия, $As|_f^{fs}$ и $Ds|_{fs}^{f}$ — скон-

струированные алгоритмы сжатия и декомпрессии соответственно, $\vec{p}$ — вектор рекомендуемых вероятностей использования базовых алгоритмов;

- $B_{20}^{0}|_{A,{}_hR}^{{}_hR}$ — добавление алгоритма A в список ${}_hR$ с инкрементом h;
- $B_{21}^{0}|_{As|_f^{fs},f}^{fs}$ — выполнение алгоритма $As|_f^{fs}$ (преобразователем алгоритмов сжатия), где fs — файлы после сжатия;
- $B_{22}^{0}|_{{}_hR}^{{}_h\hat{R}}$ — сортировка алгоритмов по показателю эффективности;
- $B_{23}^{0}|_{{}_h\hat{R}}^{{}_h\hat{\hat{R}}}$ — удаление из списка ${}_h\hat{R}$ половины алгоритмов с худшими показателями $((h\leftarrow\downarrow\hat{\hat{R}} = (h\leftarrow\downarrow\hat{R}):2)$;
- $B_{24}^{0}|_{{}_s\hat{R}_1}^{K}$, $B_{25}^{0}|_{K}^{r}$, $B_{26}^{0}|_{K,r}^{\tilde{s}_{\max}}$, $B_{27}^{0}|_{K}^{s_{\min}}$, $B_{28}^{0}|_{K,\tilde{s}_{\max},s_{\min}}^{\vec{l}}$, $B_{29}^{0}|_{K,\vec{l}}^{\vec{Pq}}$ — алгоритмы, необходимые для кластеризации, расчета вектора рекомендуемых вероятностей и выбора наилучшего алгоритма сжатия;
- $B_{30}^{0}|_{\vec{Pq}}^{\vec{Pk}}$ — расчет $\vec{Pk}$ (вектора рекомендуемых вероятностей использования кластера, как образца для синтеза адаптивного алгоритма);
- $B_{31}^{0}|_{K,r,\vec{Pk},\vec{l}}^{\vec{p}}$ — расчет рекомендуемой вероятности применения базового алгоритма при условии выбора r-го кластера;
- $B_{32}^{0}|_{Ad,Dd,\vec{p}}^{rez,Ad,Dd,\vec{p}}$ — выполнение анализа, где $rez = 1$, если степень сжатия достаточная, и $rez = 0$ в противном случае, Ad — налучший по показателю эффективности алгоритм сжатия, Dd — соответствующий ему алгоритм декомпрессии.

Значение атрибута семантики для нетерминалов: $\varphi\leftarrow\downarrow\sigma$ — начальный символ — адаптация; $\varphi\leftarrow\downarrow\delta$ — синтез; $\varphi\leftarrow\downarrow\chi$ — выполнение с измерениями; $\varphi\leftarrow\downarrow\theta$ — анализ, $\varphi\leftarrow\downarrow\xi$ — расчет рекомендуемых вероятностей, $\varphi\leftarrow\downarrow\lambda$ — контроль качества адаптации.

Аксиоматика Λ_{14} содержит аксиомы и дополнения, задающие начальные значения атрибутов и входных параметров в начале вывода:

- для правил с аксиомой в левой части принимаются: $Nv = 100$ — количество формируемых пар алгоритмов сжатия и декомпрессии, $n = 11$ — количество базовых алгоритмов, ${}_hR$ и ${}_hRd$ — соответственно списки алгоритмов сжатия и декомпресси пустые, $h = 0$;
- начальные значения элементов вектора вероятностей принимаются $\vec{p} = \langle 0.5, 0.2, 0.5, 0.3, 0.05, 0.1, 0.2, 0.25, 0.4, 0.5\rangle$, а затем рассчитываются на основании анализа;
- для сжатия используются файлы из банка данных из 840 файлов размером 40…1000 Кбайт и общим объемом 367.2 Мб, $f =$ «*D*:*DB*».

Приведем правила подстановки частичной аксиоматики Λ_{14}.

На основании отношения правила подстановки s_1 формируется текущая форма в виде композиции абстрактных операторов $\delta\cdot\chi\cdot\vartheta\cdot\sigma$ с уже определенным вектором $\vec{p}$:

$$s_1 = \langle \sigma|_{Nv,{}_hR,{}_hRd,f\ \vec{d}}^{Ad,Dd} \to \delta|_{{}_hR,{}_hRd,Nv,f,\vec{p}}^{{}_hR,{}_hRd} \cdot \chi|_{{}_hR,f,1}^{{}_hR} \cdot \vartheta|_{{}_hR,{}_hRd}^{\vec{p},Ad,Dd} \cdot \sigma|_{Nv,{}_hR,{}_hRd,f}^{Ad,Dd,\vec{p}}\rangle;$$

$$\tau_1 g_1 = \langle d_3 := 1, d_1 := 0\rangle.$$

Согласно отношению s_2 завершается процесс адаптации:

$$s_2 = \langle \sigma |^{Ad, Dd, \vec{p}}_{Nv, {}_hR, {}_hRd, f\ \vec{d}} \to \varepsilon \rangle, \quad \tau_1 g_2 = \langle d_1 := 1, d_2 := 0 \rangle.$$

Рекурсивное применение следующего отношения позволяет сконструировать Nv пар алгоритмов сжатия и декомпрессии. Алгоритмы сжатия заносятся в список ${}_hR$, а алгоритмы декомпрессии в ${}_hRd$:

$$s_3 = \langle \delta |^{{}_hR, {}_hRd}_{{}_hR, {}_hRd, Nv, f, \vec{p}\ \vec{d}} \to B^0_{19} |^{As|^{fs}_f, Ds|^f_{fs}}_{f, \vec{p}} \cdot B^0_{20} |^{{}_hR}_{As|^{fs}_f, {}_hR} \cdot$$
$$\cdot B^0_{20} |^{{}_hRd}_{Ds|^f_{fs}, {}_hRd} \cdot \delta |^{{}_hR, {}_hRd}_{{}_hR, {}_hRd, Nv, f, \vec{p}} \rangle,$$

$$\tau_1 g_3 = \langle \div (h / Nv = 0, 2, (d_3 := 0, d_4 := 1)), \div (h / Nv > 0, 2, (d_3 := 1, d_4 := 0)) \rangle.$$

Отношение s_4 необходимо для завершения процесса синтеза:

$$s_4 = \langle \delta |^{{}_hR, {}_hRd}_{{}_hR, {}_hRd, Nv, f, \vec{p}\ \vec{d}} \to \varepsilon \rangle, \quad \tau_1 g_4 = \langle d_4 := 0, d_5 := 1 \rangle.$$

Следующее отношение позволяет выполнить процессы сжатия сконструированными алгоритмами. Измерение степени сжатия задано соответствующими операциями над атрибутами

$$s_5 = \langle \chi |^{{}_hR}_{{}_hR, f, i\ \vec{d}} \to B^0_{18} |^{As_i|^{fs}_f}_{{}_hR} \cdot B^0_{21} |^{fs}_{As_i|^{fs}_f, f} \cdot \chi |^{{}_hR}_{{}_hR, f, i} \rangle;$$

$$\tau_1 g_5 = \langle s_i := \nabla (f, fs), i := i + 1, \div (i > h, 2, (d_6 := 1, d_5 := 0)) \rangle.$$

Шестое правило предназначено для завершения процессов выполнения алгоритмов сжатия

$$s_6 = \langle \chi |^{{}_hR}_{{}_hR, f, i\ \vec{d}} \to \varepsilon \rangle; \quad \tau_1 g_6 = \langle d_7 := 1, d_6 := 0 \rangle.$$

Согласно отношению s_7 в текущей форме абстрактный оператор ϑ заменяется на композицию нетерминалов ξ и λ. Выбор наилучшего алгоритма сжатия и соответствующего ему алгоритма декомпрессии (*best* — номер наилучшего алгоритма сжатия в списке ${}_hR$) определяется операциями над атрибутами

$$s_7 = \langle \vartheta |^{p, Ad, Dd}_{{}_hR, {}_hRd\ \vec{d}} \to \xi |^{\bar{p}}_{{}_hR} \cdot \lambda |^{rez, Ad, Dd, \bar{p}}_{Ad, Dd, \bar{p}} \rangle,$$

$$\tau_1 g_7 = \langle best := \Diamond ({}_hR), Ad = \hbar ({}_hR, best), Dd = \hbar ({}_hRd, best), d_7 := 0, d_8 := 1 \rangle.$$

На основании отношения подстановки s_8 нетерминал ξ в текущей форме заменяется на композицию терминалов, которые представляют процесс расчета рекомендуемых вероятностей использования базовых алгоритмов:

$$s_8 = \langle \xi |^{\vec{p}}_{{}_hR\ \vec{d}} \to B^0_{22} |^{h, s\hat{R}}_{{}_hR} \cdot B^0_{23} |^{{}_s\hat{R}_1}_{h, s\hat{R}} \cdot B^0_{24} |^{K}_{{}_s\hat{R}_1} \cdot B^0_{25} |^{r}_{K} \cdot B^0_{26} |^{\breve{s}_{\max}}_{K, r} \cdot B^0_{27} |^{s_{\min}}_{K} \cdot$$
$$\cdot B^0_{28} |^{\vec{l}}_{K, \breve{s}_{\max}, s_{\min}} \cdot B^0_{29} |^{\overrightarrow{Pq}}_{K, \vec{l}} \cdot B^0_{30} |^{\overrightarrow{Pk}}_{\overrightarrow{Pq}} \cdot B^0_{31} |^{\vec{p}}_{K, r, \overrightarrow{Pk}, \vec{l}} \rangle,$$
$$\tau_1 g_8 = \langle d_8 := 0, d_9 := 1 \rangle.$$

На основании следующего отношения подстановки выполняется анализ достигнутого качества сжатия и необходимости дальнейшей адаптиции:

$$s_9 = \langle \lambda \,|^{rez, Ad, Dd, \vec{p}}_{Ad, Dd, \vec{p}} \; \vec{d} \to B^0_{32}|^{rez, Ad, Dd, \vec{p}}_{Ad, Dd, \vec{p}} \rangle,$$

$$\tau_1 g_9 = \langle \div(rez = 1, 2, (d_1 := 0, d_2 := 1)),\ \div(rez = 0, 2, (d_1 := 1, d_2 := 0)) \rangle,\ d_9 := 0 \rangle.$$

Рассмотри конструкцию процесса адаптации, сформированной в результате операции полного вывода

$$l = \prod_{j=1}^{100} (B^0_{19}|^{As|^{fs}_f, Ds|^{fs}_f}_{f, \vec{p}} \cdot B^0_{20}|^{{}_hR}_{As|^{fs}_f, {}_hR} \cdot B^0_{20}|^{{}_hRd}_{Ds|^{f}_{fs}, {}_hRd}) _j \cdot \prod_{i=1}^{100} (B^0_{18}|^{As_i|^{fs}_f}_{{}_hR} \cdot B^0_{21}|^{fs}_{As_i|^{fs}_f, f}) \cdot$$
$$\cdot B^0_{22}|^{h,s\hat{R}}_{{}_hR} \cdot B^0_{23}|^{s\hat{R}_1}_{h,s\hat{R}} \cdot B^0_{24}|^{K}_{s\hat{R}_1} \cdot B^0_{25}|^{r}_{K} \cdot B^0_{26}|^{\breve{s}_{\max}}_{K,r} \cdot B^0_{27}|^{s_{\min}}_{K} \cdot B^0_{28}|^{\vec{l}}_{K, \breve{s}_{\max}, s_{\min}} \cdot$$
$$\cdot B^0_{29}|^{\vec{Pq}}_{K, \vec{l}} \cdot B^0_{30}|^{\vec{Pk}}_{\vec{Pq}} \cdot B^0_{31}|^{\vec{p}}_{k, r, \vec{Pk}, \vec{l}} \cdot B^0_{23}|^{res, Ad, Dd, \vec{p}}_{Ad, Dd, \vec{p}},$$

которая отображает основу процесса адаптации. Неотъемлемым дополнением конструктивного процесса являются операции над атрибутами в процессе вывода.

Реализация КПС адаптера алгоритмов сжатия представляет собой пару: алгоритм сжатия, адаптированный к заданному набору данных на конкретном компьютере, и соответствующий ему алгоритм декомпрессии. Пара наилучших алгоритмов сжатия и декомпрессии определяется атрибутами *Ad* и *Dd* соответственно.

В результате реализации адаптера алгоритмов сжатия и соответствующих двойственного конструктора и преобразователя в программной реализации [7] сформированы адаптивные алгоритмы. Например, сконструированный алгоритм сжатия $B^0_{17}|^{M}_{f} \cdot B^0_1|^{M}_{M} \cdot B^0_4|^{Ms}_{M} \cdot B^0_8|^{Ms}_{M} \cdot B^0_7|^{Ms}_{M} \cdot B^0_6|^{Ms}_{M} \cdot B^0_5|^{Ms}_{M} \cdot B^0_6|^{Ms}_{M} \cdot B^0_8|^{Ms}_{M} \cdot B^0_9|^{fs}_{M, fs}$ является конкатенацией базовых алгоритмов чтения данных из файла, предобработки arcDelta и RLE, сжатия методами GZip, arcLZMA, arcBZip, arcDeflate, arcBZip, GZip и сохранения сжатых данных в новый файл соответственно. Алгоритм декомпрессии соостоит из соотвествующих обратных алгоритмов

$$(B^{-1}_{17})^0|^{f}_{M} \cdot (B^{-1}_1)^0|^{M}_{M} \cdot (B^{-1}_4)^0|^{M}_{Ms} \cdot (B^{-1}_8)^0|^{M}_{Ms} \cdot (B^{-1}_7)^0|^{M}_{Ms} \cdot$$
$$\cdot (B^{-1}_6)^0|^{M}_{Ms} \cdot (B^{-1}_5)^0|^{M}_{Ms} \cdot (B^{-1}_6)^0|^{M}_{Ms} \cdot (B^{-1}_8)^0|^{M}_{Ms} \cdot (B^{-1}_9)^0|^{M, fs}_{fs}.$$

ЗАКЛЮЧЕНИЕ

Предложенная модель адаптации алгоритмов сжатия представляет собой совокупность трех КПС: двойственного конструктора алгоритма сжатия (компрессии и декомпрессии), преобразователя сконструированного алгоритма в конструктивный процесс сжатия и непосредственно КПС адаптера. Ввиду того, что адаптер использует результаты преобразователя, а последний, в свою очередь, — двойственного конструктора, метод поэтапного проектирования КПС (определение специализации, интерпретация, конкретизация и реализация) дополняется средствами связывания КПС.

Разработанная модель является теоретическим обоснованием, представленного в [7, 15] процесса адаптации алгоритмов сжатия.

Описанные КПС существенно расширяют возможности моделирования процесса адаптации. Так, один и тот же конструктор связан с различными адаптерами, что позволяет получить адаптацию по времени и степени сжатия; адаптер использует разные преобразователи алгоритмов сжатия. В настоящей статье представлен конструктор, который формирует алгоритмы сжатия без потерь информации, однако можно определить специализацию конструктора для сжатия и с потерями.

На этапе конкретизации можно задавать специфические алгоритмы для анализа, более того, систему анализа задавать отдельной КПС, также определить, как будут транслироваться и исполнятся алгоритмы. В данной статье разработана структура преобразователя алгоритмов сжатия в процесс их выполнения без детализации исполнителя.

Перспективна разработанная модель двойственного конструктора. Данный подход позволяет конструировать идентичные и отличающиеся конструкции, как, например, конструктивные процессы искусственной речи и сопровождающего ее видеоизображения.

СПИСОК ЛИТЕРАТУРЫ

1. Гинзбург С. Математическая теория контекстно-свободных языков. — М.: Книга по требованию, 2012. — 286 с.
2. Фу К. Структурные методы распознавания образов. — М.: Мир, — 1977. — 318 с.
3. Шинкаренко В.И., Ильман В.М. Конструктивно-продукционные структуры и их грамматические интерпретации. I. Обобщенная формальная конструктивно-продукционная структура // Кибернетика и системный анализ. — 2014. — **50**, № 5. — С. 8–16.
4. Шинкаренко В.И., Ильман В.М., Скалозуб В.В. Структурные модели алгоритмов в задачах прикладного программирования. I. Формальные алгоритмические структуры // Кибернетика и системный анализ. — 2009 — № 3. — С. 3–14.
5. Шинкаренко В.И., Ильман В.М. Конструктивно-продукционные структуры и их грамматические интерпретации. II. Уточняющие преобразования // Кибернетика и системный анализ. — 2014. — **50**, № 6. — С. 15–28.
6. Сэломон. Д. Сжатие данных, изображения и звука. — М.: Техносфера, 2004. — 368. с.
7. Шинкаренко В.И., Кроль Г.Г., Васецкий Е.Г., Мажара Т.Н. Структурная адаптация алгоритмов сжатия данных на метаалгоритмической основе // Искусственный интеллект. — 2009. — № 4. — С. 104–111.
8. Миано Дж. Форматы и алгоритмы сжатия изображений в действии. — М.: Триумф, 2003. — 336 с.
9. Ільман В.М., Шинкаренко В.І. Структурний підхід до проблеми відтворення граматик // Проблемы программирования. — 2007 — № 1. — С. 5–16.
10. Шинкаренко В.И. Знание-ориентированный подход к адаптации алгоритмов // Искусственный интеллект. — 2008. — № 3. — С. 388–397.
11. Delta encoding in HTTP / J. Mogul, B. Krishnamurthy, F. Douglis, A. Feldmann, Y. Goland, A. van Hoff, D. Hellerstein // RFC 3229. — 2002. — 49 p. — http://tools.ietf.org/html/rfc3229.
12. The Base16, Base32, and Base64 Data Encodings / S. Josefsson, Ed. // RFC 3548. — 2003. — 13 p. — http://tools.ietf.org /html /rfc3548.
13. Ватолин Д., Ратушняк А., Смирнов М., Юкин В. Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео. — М.: Диалог-МИФИ, 2003. — 384 с.
14. Растригин Л.А. Адаптация сложных систем. — Рига: Зинатне, 1981. — 375 с.
15. Шинкаренко В.И., Кроль Г.Г., Васецкий Е.Г. Методы и средства структурной адаптации алгоритмов на метаалгоритмической основе // Искусственный интеллект. — 2009. — № 3. — С. 105–113.

Поступила 26.01.2015