УДК 519.681

В. Г. Акуловский, кандидат технических наук,
доцент кафедры информационных систем
и технологий Университета таможенного дела
и финансов
В. В. Костенко, старший преподаватель
кафедры информационных систем
и технологий Университета таможенного дела
и финансов
В. В. Полищук, старший преподаватель
кафедры информационных систем
и технологий Университета таможенного дела
и финансов
В. Н. Пономарёв, старший преподаватель
кафедры информационных систем
и технологий Университета таможенного дела
и финансов

СОСТАВ, СТРУКТУРА И СВОЙСТВА ДАННЫХ В АЛГОРИТМАХ, ОПИСАННЫХ С ПОМОЩЬЮ КОМПОЗИЦИОННЫХ СХЕМ

Приведены основные характеристики алгебраического аппарата. Введены определения алгоритма и подсистем, проиллюстрирована возможность перехода от алгоритма к описанию образующих его подсистем. Показаны состав и структура данных, специфицированных на входах и выходах Д-операторов, входящих в композиционные схемы, посредством которых описываются алгоритмы. Продемонстрирована возможность описания входных и выходных данных и доказаны их свойства. Определено понятие информационных связей и показаны свойства данных, образующих эти связи.

Ключевые слова: *алгебра алгоритмов; Д-операторы; композиционные схемы; свойства и структура данных.*

Наведено основні характеристики алгебраїчного апарату. Введено визначення алгоритму й підсистем та проілюстровано можливість переходу від алгоритму до опису підсистем, що утворюють його. Показано склад і структуру даних, специфікованих на входах і виходах Д-операторів, що входять в композиційні схеми, за допомогою яких описуються алгоритми. Продемонстровано можливість опису у вхідних і вихідних даних та доведено їхні властивості. Визначено поняття інформаційних зв'язків та показано властивості даних, що утворюють ці зв'язки.

Ключові слова: *алгебра алгоритмів; Д-оператори; композиційні схеми; властивості і структура даних.*

There are showed the algebraic apparatus elements designed for a consistent description of the pots and control algorithms. In particular, showed data definition, D-operators and the operations defined on them set. There are showed the possibility of description of algorithms in the form of compositional schemes.

The purpose of the work in connection with the key role of data in developing the algorithm is in the study of composition, structure and properties of the data specified in the proposed algorithms.

The composition and structure of data processed by the algorithm interacting with the external environment, as well as the availability of information links in them. There are demonstrated the specification ability information links between D-operators, forming a composition diagram, by describing the algorithms. While specified not only linking the data but all the data "sources" and data "receivers" set in one mapping. The data using restriction has allowed to prove some properties of the input, output, sourcing, resulting and linking data.

The obtained properties and specifications of information links in the algorithms can be used as a means of verification of its correctness from the point of view of data processing. Such control can be carried out at all levels of the decomposition algorithm.

Future analysis of the information relationships and identifying new properties of these algorithms is a promising direction for further research to address problems of information complexity of algorithms and their optimizing transformations.

Key words: *algebra of algorithms; D-operators; compositional schemas; properties and data structure.*

Постановка проблемы. На процесс проектирования алгоритмов и программ принципиально важное влияние оказывают данные [1–3]. Более того, в работах [4; 5] утверждается, что информация о данных для понимания программ важнее, чем информация об управлении. Роль данных и их влияние на процесс проектирования алгоритмов в рамках алгебраических аппаратов изучены недостаточно.

Анализ последних исследований и публикаций. Для решения задачи формализованного проектирования алгоритмов в результате модификации известной модели ЭВМ Глушкова [6] был разработан [7–10] алгебраический аппарат (алгебра алгоритмов с данными).

Данными в алгебре называется пара $D = \langle N_j, 3 \rangle$, где $3 = \langle z_1, ..., z_n \rangle$ – кортеж последовательно расположенных значений, определяющий в каждый момент времени текущее состояние данных, носитель которого N_j однозначно идентифицируется индексом $j \in J$, который интерпретируется как адрес, принадлежащий множеству доступных адресов J. Носитель данных отождествляется с памятью и внешними устройствами (дальше – ВУ) и называется аппаратной средой, а множество данных, хранимых носителем, обозначено D^{AC}. В каждый момент времени носитель содержит (хранит) некоторые (текущие) значения данных (в частности, эти значения могут быть неопределенными), которые изменяются в результате выполнения Д-операторов. С другой стороны, носитель является источником и приемником данных, обрабатываемых алгоритмом.

На множестве данных определены операции **укрупнения** $*\{D_1, D_2 ..., D_k\} = D$ и **детализации** данных $\overline{*}\ \{D_1, D_2 ..., D_k\}$.

Основой для построения алгоритмов являются Д-операторы, на входе и выходе которых специфицированы множества обрабатываемых данных. Определим актуальные для данной работы типы введенных Д-операторов.

Определение 1. Д-оператор $(D)O(D')$ обрабатывает множество и только множество данных $D^{\text{o}} \subseteq D^{AC}$ в результате выполнения конечной последовательности действий (операций), называемых функциональностью Д-оператора. На текущем уровне детальности представления Д-оператора специфицируются входные $D \subseteq D^{\text{o}}$ и изменяемые в результате выполнения Д-оператора выходные $D' \subseteq D^{\text{o}}$ данные, актуальные для конкретного уровня. Д-оператор, не изменяющий данных, называется тождественным и обозначается *Z*. Д-оператор, в результате выполнения которого вычислительный процесс переходит в неопределенное состояние, называется неопределенным и обозначается *N*.

Предложенное определение Д-операторов позволяет на каждом шаге проектирования рассматривать только актуальные для конкретного шага данные D и D', абстрагируясь от остальных (неактуальных на данном уровне представления) данных.

Отметим, что при рассмотрении специфицированных данных в виде семейств множеств Д-оператор может записываться в виде $(D_1,..., D_p)O(D'_1,..., D'_s)$.

Второй тип Д-оператора – $(D^{\pi})P(\alpha)$, называемый предикатом, представляет собой *n*-местную (в общем случае) логическую функцию $P: D^{\pi} \to \alpha \in E_2$, где D^{π} – график некоторого *n*-го отношения; $E_2 = \{1,0\}$; α – двузначное логическое условие, которое анализирует множество данных D^{π} и продуцирует логическое условие α, характеризующее отношение между текущими значениями этих данных. В результате выполнения предиката значение данных D^{π} не изменяется.

На множестве логических условий определены следующие известные булевы операции: конъюнкция ($\wedge$), дизъюнкция ($\vee$) отрицание $\overline{\alpha}$ с известными таблицами истинности [11].

На множестве Д-операторов определены операции, из числа которых приведем следующие.

Композиция Д-операторов $(\dot{D})\dot{O}(\dot{D}')*(\tilde{D})\tilde{O}(\tilde{D}')$ означает последовательное выполнение сначала Д-оператора $(\dot{D})\dot{O}(\dot{D}')$, затем Д-оператора $(\tilde{D})\tilde{O}(\tilde{D}')$.

Рассмотрим структуру данных, обрабатываемых парой Д-операторов, связанных операцией композиции $(\dot{D})\dot{O}(\dot{D}')*(\tilde{D})\tilde{O}(\tilde{D}')$.

Особо отметим, что используемые ниже теоретико-множественные операции определены на множествах носителей данных.

Принципиально важными особенностями обработки данных такой конструкцией является следующее.

Во-первых, обработка данных (или их части) может выполняться “коллективно”, то есть обработка данных, начатая одним Д-оператором, может быть продолжена следующим за ним Д-оператором.

Во-вторых, обработка данных (или их части) может выполняться “индивидуально”, то есть каждый из Д-операторов в общем случае обрабатывает и продуцирует некоторое множество данных, на значения которых выполнение других Д-операторов не влияет.

В-третьих, наряду с решением глобальной задачи по преобразованию некоторых входных глобальных данных в выходные каждый Д-оператор может решать некоторую локальную задачу, обрабатывая локальные данные, то есть в общем случае на входе и выходе Д-оператора $(D)O(D')$ специфицированы $D^{Л} \subseteq D$ и $D'^{Л} \subseteq D'$.

В связи с указанными аспектами взаимодействия Д-операторов, входящих в композицию, введем следующее определение.

Определение 2. В композиции Д-операторов $(\dot{D})\dot{O}(\dot{D}')*(\tilde{D})\tilde{O}(\tilde{D}')$ на входе и выходе $(\dot{D})\dot{O}(\dot{D}')$ специфицированы $\dot{D} = \dot{D}'' \cup \dot{D}^{Л}$ и $\dot{D}' = \dot{D}^{СБ} \cup \dot{D}'^{Л} \cup D^{СВ}$, а на входе и выходе $(\tilde{D})\tilde{O}(\tilde{D}')$ – $\tilde{D} = \tilde{D}^{СБ} \cup \tilde{D}^{Л} \cup D^{СВ}$ и $\tilde{D}' = \tilde{D}''' \cup \tilde{D}'^{Л}$, где $\dot{D}''$, $\tilde{D}'''$, $\dot{D}^{СБ}$, $\tilde{D}^{СБ}$ – глобальные, $\dot{D}^{Л}$, $\dot{D}'^{Л}$, $\tilde{D}^{Л}$, $\tilde{D}'^{Л}$ – локальные данные. $\dot{D}^{СБ}$ и $\tilde{D}^{СБ}$ – такие собственные данные, которые $\tilde{D}^{\text{СБ}}$ не изменяются Д-оператором $(\dot{D})\dot{O}(\dot{D}')$, а $\dot{D}^{СБ}$ – $(\tilde{D})\tilde{O}(\tilde{D}')$. $D^{СВ}$ – такие связывающие данные, что $D^{СВ} = \dot{D}' \cap \tilde{D}$. Локальные данные, специфицированные на входе и выходе Д-оператора, обрабатываются только конкретным Д-оператором и не используются ни одним другим Д-оператором на текущем уровне представления. Любые множества данных на входах и выходах Д-операторов, но не все сразу, могут быть пустыми.

Определение 2 проиллюстрируем рис. 1 и поясним следующим образом.

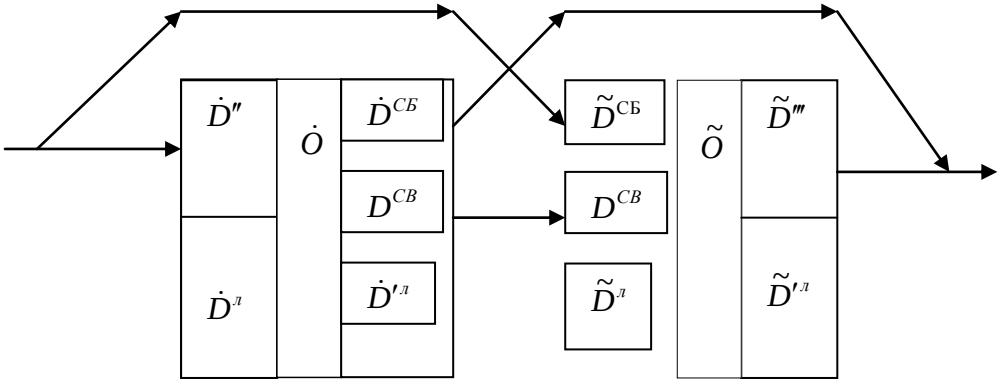

Рис. 1. Данные, специфицированные на входе и выходе Д-операторов

Собственные данные $\dot{D}^{СБ}$ являются результатом выполнения только Д-оператора $(\dot{D})\dot{X}(\dot{D}')$, $\tilde{D}^{СБ}$ – обрабатываются только Д-оператором $(\tilde{D})\tilde{X}(\tilde{D}')$, а связывающие данные $D^{СВ}$ обрабатываются совместно двумя Д-операторами.

Кроме рассмотренной на множестве Д-операторов операции, определены следующие операции:

р-дизъюнкция

$$[(D^{\pi})P(\alpha)]((D_1)O_1(D_1')*(D_2)O_2(D_2')) = \\ = \begin{cases} (D_1)O_1(D_1'), \text{ если } \alpha = 1; \\ D_2)O_2(D_2'), \text{ если } \alpha = 0, \end{cases}$$

в результате выполнения которой выбирается один из двух возможных Д-операторов в соответствии со значением логического условия α;

р-итерация $[(D^{\pi})P(\alpha)]\{(D)O(D')\}$ осуществляет циклическое выполнение Д-оператора $(D)O(D')$ при $\alpha = 1$ и завершается в противном случае. В частном случае эта операция, записанная в виде $\langle 1 \rangle \{(D)O(D')\}$, где 1 – тождественно истинное условие, реализует "бесконечный" цикл.

Таким образом, алгебра алгоритмов – это система алгоритмических алгебр $САА\backslash Д ::= \langle \{X, L, D\}; \Omega \rangle$, где X – множество Д-операторов; L – множество логических условий; D – множество данных; Ω – сигнатура алгебры, включающая операции Ω_1, определенные на множестве X; Ω_2 – определенные на множестве L; Ω_3 – определенные на этом множестве D.

Поставленную задачу будем решать, используя, обобщая и дополняя результаты, полученные в работах [12–16].

Цель статьи – изучение состава, структуры и свойств данных, специфицируемых в разрабатываемых алгоритмах.

Изложение основного материала. Алгоритмом будем называть Д-оператор, функциональность которого (см. определение 1) обеспечивает решение некоторой задачи, то есть преобразование некоторых исходных данных в результирующие (являющиеся результатом функционирования алгоритма).

В работах [6; 11] показано, что в рамках алгебраического аппарата любые Д-операторы (в частности, алгоритмы) записываются в виде регулярных схем (РС), представляющих собой суперпозиции Д-операторов и операций алгебры.

Учитывая, что на функциональность Д-операторов не накладываются ограничения, будем рассматривать операции алгебры как специальный тип Д-операторов. То есть операции р-дизъюнкции

$$[(D^{\pi})P(\alpha)]((D_1)O_1(D_1')*(D_2)O_2(D_2'))$$

соответствует Д-оператор $(D)O^{\partial}(D')$, где $D = (D^{\pi} \cup D_1 \cup D_2)$, $D' = (D'_1 \cup D'_2)$, а р-итерации $[(D^{\pi})P(\alpha)]\{(D)O(D')\}$ – Д-оператор $(D)O^{u}(D')$,

где $$D = (D^{\pi} \cup D), \; D' = \dot{D}'. \qquad (1)$$

С использованием структуры данных, приведенных в определении 2, показанной на рис. 1, в работах [17–18] доказана реализуемость восходящей и нисходящей стратегий проектирования алгоритма. Это позволяет записывать Д-оператор, декомпозируя его, в виде суперпозиции Д-операторов, включающих введенные выше:

$$(D)O(D') = (D_1)O_1(D'_1)*(D_2)O_2(D'_2)*...*(D_k)O_k(D'_k), \qquad (2)$$

где $(D_1)O_1(D'_1)$, $(D_2)O_2(D'_2)$, …, $(D_k)O_k(D'_k)$ – производные Д-операторы, полученные в результате декомпозиции исходного.

Полученные таким образом схемы называются композиционными (КС) и удобны в использовании при дальнейшем изложении.

Рассмотрение данных, обрабатываемых алгоритмом, начнем с описания взаимодействия алгоритма с аппаратной средой, воспользовавшись результатами, полученными в работе [19]. Учитывая, что аппаратная среда представляет собой память и внешние устройства и является источником и приемником данных, введем следующее определение.

Определение 3. Множество данных D^{AC}, хранимых носителем, представляет собой $D^{AC} = (D^{П} \cup D^{ВУ})$, где $D^{П}$ – множество данных, хранимых в памяти, $D^{ВУ}$ – множество данных, хранимых ВУ. Множество данных $D^{ВУ} = D^{R} \cup D^{W}$, где D^{R} – множество данных, читаемых (переносимых, копируемых) с ВУ в память и называемых вводимыми, а D^{W} – множество данных, записываемых (переносимых, копируемых) из памяти на ВУ и называемых выводимыми. Вводимые и выводимые данные специфицируются соответственно на входе $D^{R} \subseteq D$ и выходе $D^{W} \subseteq D'$ Д-оператора и в общем случае $D^{R} \cap D^{W} \neq \varnothing$, а в частных случаях множества D^{R} и/или D^{W} могут быть пустыми $(D^{R} = \varnothing$ и/или $D^{W} = \varnothing)$, то есть вводимые и/или выводимые данные могут отсутствовать. Исходные (определенные, инициированные) данные, имеющие заданные начальные значения, такие, что $D^{ИСХ} \subseteq D^{П}$, $D^{ИСХ} \subseteq D$ и $D^{R\_исх} \subseteq D^{R}$. Результирующие данные такие, что $D^{W\_рез} \subseteq D^{W}$, и после вывода эти данные более не вводятся.

Из определения следует, что в общем случае Д-оператор со специфицированными данными $(D)O(D')$ такой, что

$$D^{ИСХ}, \; D^{R} \subseteq D \;\text{ и }\; D^{W} \subseteq D', \qquad (3)$$

а в частных случаях – $(D)O(D^W)$, $(D^R)O(D')$ и т. д. В качестве вводимых и выводимых могут выступать исходные и результирующие данные.

Алгоритм и обрабатываемые им данные определим следующим образом.

Определение 4. Верхний (наиболее общий) уровень описания произвольного алгоритма представляет собой Д-оператор $(D)A(D')$, преобразующий глобальные для решаемой задачи, специфицированные на его входе и выходе данные, такие, что $D = D^{ИСХ} = D^{исх} \cup D^{R\_исх}$, $D' = D^{РЕЗ} = D^{W\_рез}$, где $D^{РЕЗ} \neq \varnothing$, а в частных случаях $D^{исх} = \varnothing$ и/или $D^{R\_исх} = \varnothing$. Иначе говоря, алгоритм записывается в виде $(D^{ИСХ})A(D^{РЕЗ})$.

В данном случае будем рассматривать более распространенную нисходящую стратегию проектирования.

Первый шаг проектирования в рамках нисходящей стратегии проектирования определяет архитектуру программной системы, которая играет ключевую роль, так как от её реализации в существенной мере зависят все последующие этапы и, в конечном счете, качество программного продукта. На этом этапе решаются две взаимосвязанные основные задачи:

1) определяется состав подсистем, образующих алгоритм, и специфицируются обрабатываемые и продуцируемые этими подсистемами данные;

2) специфицируются взаимосвязи между этими подсистемами.

Исходя из этих задач, подсистему определим следующим образом.

Определение 5. Подсистемой назовем Д-оператор $(D_i)P_i(D_i')$, реализующий часть функциональности алгоритма, обрабатывающий некоторую часть исходных и продуцирующий часть результирующих данных, специфицированных на его входе и выходе. Совокупность информационно связанных подсистем, образующих алгоритм, реализует всю его функциональность, обрабатывает все данные, специфицированные на его входе, и с помощью дополнительно введенных локальных данных (в дальнейшем – дополнительных данных) продуцирует все данные, специфицированные на его выходе. Значения дополнительно введенных данных на момент начала выполнения алгоритма не определены.

Воспользовавшись соотношением (2), запишем алгоритм с учетом определений 4 и 5 в виде следующей КС:

$$\begin{aligned}(D^{ИСХ})A(D^{РЕЗ}) = ({}_1D_1^{ИСХ}, {}_1D_1)_1P_1({}_1D_1^{РЕЗ}, {}_1D_1') * \ldots \\ \ldots * ({}_1D_k^{ИСХ}, {}_1D_k)_1P_k({}_1D_k^{РЕЗ}, {}_1D_k'),\end{aligned} \quad (4)$$

где левый нижний индекс обозначает шаг разработки, а любые ${}_1D_i$ и ${}_1D_i'$ – дополнительно введенные данные, упомянутые в определении 5.

Таким образом, программная система разбивается на подсистемы, а все исходные и результирующие данные, в соответствии с определением 5, "распределяются" между подсистемами и для них выполняются следующие соотношения:

$$D^{\text{ИСХ}} = \bigcup_{i=1}^{n} {}_1D_i^{\text{ИСХ}} \ , \ D^{\text{РЕЗ}} = \bigcup_{i=1}^{n} {}_1D_i^{\text{РЕЗ}} \ . \qquad (5)$$

Поскольку подсистемы, образующие программную систему, в соответствии с определением 5, взаимосвязаны, расширим понятие связи между Д-операторами посредством следующего определения.

Определение 6. Д-операторы $(D_i)O_i(D_i')$ и $(D_j)O_j(D_j')$ (где $i<j$), входящие в КС:

$$(D)O(D') = (D_1)O_1(D_1')*...*(D_i)O_i(D_i')*(D_{i+1})O_{i+1}(D_{i+1}')*...$$
$$...*(D_j)O_j(D_j')*...*(D_k)O_k(D_k'),$$

прямо связаны, если для них выполняется соотношение $D_i' \cap D_j \neq \varnothing$, а данные ${}_i\vec{D}_j = D_i' \cap D_j$ – прямо связывающие эти Д-операторы, или прямые связи. В случае, когда $D_i' \cap D_{i+1} \neq \varnothing$, Д-операторы $(D_i)O_i(D_i')$ и $(D_{i+1})O_{i+1}(D_{i+1}')$ связаны непосредственно, а данные ${}_i\vec{D}_{i+1}$ – непосредственно связывающие эти Д-операторы, или непосредственные связи. Все связи Д-оператора $(D_i)O_i(D_i')$ со всеми следующими за ним и всеми предшествующими ему Д-операторами обозначим $\vec{D}_i^{CB} = \bigcup_{j=i+1}^{k} {}_i\vec{D}_j$ и $\overleftarrow{D}_i^{CB} = \bigcup_{j=1}^{i-1} {}_j\overleftarrow{D}_i$ и назовем правыми и левыми его связями. Любое из множеств ${}_i\vec{D}_s \subseteq D_i^{\text{CB}}$ и ${}_i\overleftarrow{D}_p \subseteq D_i^{\text{CB}}$ может быть пустым (${}_i\vec{D}_s = \varnothing$ и ${}_i\overleftarrow{D}_p = \varnothing$).

Из определения 6 легко увидеть, что если ${}_i\vec{D}_s = \varnothing$, то и ${}_s\overleftarrow{D}_i = \varnothing$. Будем говорить, что связывающие данные образуют информационные связи в КС.

Теперь рассмотрим структуру дополнительных данных, для чего введем ограничения на использование данных в КС.

Определение 7. Значения всех специфицированных на входе Д-оператора, входящего в КС, данных должны быть определены, а все данные, специфицированные на его выходе (за исключением результирующих), используются, то есть специфицируются на входе, по крайней мере, одного из следующих за ним Д-операторов.

Опираясь на введенное ограничение, покажем состав дополнительных данных, при обозначении которых, учитывая, что дополнительные данные, в соответствии с определением 5, локальные, будем использовать строчные буквы.

Утверждение 1. Дополнительные данные, специфицированные в КС, в общем случае включают в свой состав на входе Д-операторов ${}_1D_i^{исх}, {}_1D_i^{r\_u\_u}, {}_1D_i^{r}, \overleftarrow{D}_i^{CB}$,

на выходе: ${}_1D_{\text{i}}^{w\_p\_p}, {}_1D_{\text{i}}^{w}, \vec{D}_{\text{i}}^{CB}$ и только эти данные (за исключением случая использования Д-оператора $(D)O^{u}(D')$).

Доказательство. В соответствии с (3), в общем случае входные и выходные данные Д-оператора включают исходные и вводимые данные ($D^{исх}, D^{r} \subseteq D$), а выходные – выводимые ($D^{w} \subseteq D'$), которые, в соответствии с определением 3, в свою очередь, включают исходные вводимые (${}_1D_{\text{i}}^{r\_u\_u}$) и результирующие (${}_1D_{\text{i}}^{w\_p\_p}$) данные. Кроме того, в соответствии с определением 6, они включают связывающие данные ($\bar{D}_{\text{i}}^{CB}$ и $\vec{D}_{\text{i}}^{CB}$).

Дальнейшие рассуждения построим от противного. Предположим, что на входе Д-оператора имеют место дополнительные данные ${}_1\dot{D}_i \subseteq {}_1D_i$, отличные от перечисленных. Поскольку, в соответствии с определением 5, значения дополнительных данных не определены, то не определены и значения ${}_1\dot{D}_i$.

Предположим, что на выходе Д-оператора имеют место дополнительные данные ${}_1\dot{D}'_i \subseteq {}_1D'_i$, отличные от перечисленных. Поскольку эти данные не выводятся и не поступают на вход ни одного из следующих Д-операторов, то они не используются.

В обоих случаях пришли к противоречию с ограничением на использование данных в КС (см. определение 7), и, таким образом, состав дополнительных данных на входе и выходе Д-операторов соответствует перечисленным в утверждении.

Утверждение доказано.

Покажем наличие некоторых свойств у данных, специфицируемых в КС.

Утверждение 2. Любому первому вводу данных (за исключением ввода исходных данных) предшествует их вывод, а для выводимых данных (за исключением результирующих) выполняется соотношение ${}_1D_i^w = {}_1D_j^r \cup ... \cup {}_1D_s^r$, где $j > i, s > i$.

Доказательство построим от противного. Предположим, что ${}_1D_i^r$ вводятся до их вывода. Поскольку, в соответствии с утверждением 1, вводимые данные входят в состав дополнительных, а значения последних, в соответствии с определением 5, не определены, то введенные данные ${}_1D_i^r$ будут неопределенными.

Предположим, что указанное соотношение не выполняется.

В случае, когда ${}_1D_i^w \setminus ({}_1D_j^r \cup ... \cup {}_1D_s^r) \neq \varnothing$, часть выведенных данных не вводится и, соответственно, не используется.

В случае, когда $({}_1D_j^r \cup ... \cup {}_1D_s^r) \setminus {}_1D_i^w \neq \varnothing$, часть данных вводится без их вывода и, соответственно, вводимые данные будут неопределенными.

Поскольку во всех случаях получено противоречие с заданным ограничением, то условия утверждения выполняются.

Утверждение доказано.

Утверждение 3. У первой подсистемы в КС нет левых связей, а у последней подсистемы (за исключением случая использования Д-оператора $(D)O^{u}(D')$) на выходе специфицируются только результирующие данные.

Доказательство. Поскольку, в соответствии с определением 6, Д-оператор в КС связан с предшествующими Д-операторами, а для первого из них такие Д-операторы отсутствуют, то он лишен левых связей.

Поскольку последний в КС Д-оператор завершает работу алгоритма, то любые данные, если они специфицированы на его выходе, не будут использоваться (за исключением результирующих), что противоречит наложенному на данные ограничению.

Утверждение доказано.

Остановимся на рассмотрении Д-оператора $(D)O^{u}(D')$, который реализует операцию р-итерации (см. (1) и использование которого при описании алгоритмов показано в работе [20]. В этой работе введены алгоритмы, состоящие из подсистем: O_n – выполняет подготовительные (стартовые) операции, $O_з$ – завершающие (заключительные) операции, O_m – "тело" алгоритма. В частности, это алгоритм, характерный для информационных систем, названный непрерывно-дискретным и записанный в виде:

$$(D)A(D') = (D_n)O_n(D'_n)*\left[\left(D^{\pi}\right)P(\alpha)\right]\left\{(D_\tau)O_\tau(D'_\tau)\right\}^{*}(D_З)O_З(D'_З)$$

Алгоритм, характерный для систем управления, названный непрерывным и записанный в виде:

$$(D)A(D') = (D_n)O_n(D'_n)*\langle 1\rangle(D_m)O_m(D'_m),$$

где 1 – тождественно истинное логическое условие.

В операции р-итерации ложное значение логического условия ($\alpha = 0$), полученное за конечное число шагов, является достаточным условием для её завершения, а операция (цикла) обладает следующими свойствами:

– если $\alpha \neq 1$ на первом витке цикла, то

$$[(D^{\pi})P(\alpha)]\{(\mathrm{D})\mathrm{O}(\mathrm{D}')\} = \mathrm{Z}, \qquad (6)$$

то есть, операция вырождается в тождественный Д-оператор;

– если $\alpha \equiv 1$, то

$$[(D^{\pi})P(\alpha)]\{(\mathrm{D})\mathrm{O}(\mathrm{D}')\} = N, \qquad (7)$$

то есть, операция вырождается в неопределенный Д-оператор, что в просторечии называется “зацикливанием” вычислительного процесса;

– если на втором витке цикла $\alpha = 0$, то

$$[(D^{\pi})P(\alpha)]\{(\mathrm{D})\mathrm{O}(\mathrm{D}')\} = [(D^{\pi})P(\alpha)]((\mathrm{D})\mathrm{O}(\mathrm{D}') \vee Z), \quad (8)$$

то есть, операция вырождается в операцию р-итерации и завершится на этом витке.

Покажем условия, необходимые для завершения циклирования, обозначая равенство множеств носителей и значений данных в виде $\mathrm{D} \cong \mathrm{D}'$.

Теорема 1. Необходимыми условиями для завершения операции р-итерации $[(D^{\pi})P(\alpha)]\{(D)O(D')\}$ при более чем двух выполненных витках цикла являются $D^{\pi} \cap D' \neq \varnothing$ и $D \cap D' \neq \varnothing$.

Доказательство построим от противного, предположив, что выполняется соотношение $D^{\pi} \cap D' = \varnothing$.

На первом витке цикла предикат $(D^{\pi})P(\alpha)$ продуцирует логическое условие $\alpha = 1$, так как в противном случае, в соответствии с (6), $[(D^{\pi})P(\alpha)]\{(\mathrm{D})\mathrm{O}(\mathrm{D}')\} = Z$.

В результате выполнения тела цикла $(D)O(D')$ множество данных D^{π} не изменяется, так как для подвергшихся изменению значений данных D' выполняется $D^{\pi} \cap D' = \varnothing$, а предикат в соответствии с его определением состояния множества данных D^{π} не изменяет.

Таким образом, вне зависимости от результата выполнения выражения $D \cap D'$ второй и все последующие витки цикла будут происходить при неизменном D^{π}, то есть получим $\alpha \equiv 1$ и, в соответствии с (7), $[(D^{\pi})P(\alpha)]\{(\mathrm{D})\mathrm{O}(\mathrm{D}')\} = N$.

Предположим, что $D^{\pi} \cap D' \neq \varnothing$ и выполняется соотношение $D \cap D' = \varnothing$.

В результате выполнения тела цикла $(D)O(D')$ множество данных D^{π} изменится ($D^{\pi} \cap D' \neq \varnothing$), и мы получим множество данных $D'^{\pi} \cong D^{\pi}$.

Если на втором витке цикла предикат $(D'^{\pi})P(\alpha)$ продуцирует истинное логическое условие ($\alpha = 1$), то на втором и всех последующих шагах цикла множество данных D останется неизменным в силу $D \cap D' = \varnothing$. Это приведет к неизменности данных D', что, в свою очередь, приведет к неизменности D'^{π}. Таким образом, получаем вышерассмотренный случай $[(D^{\pi})P(\alpha)]\{(D)O(D')\} = N$.

Если на втором витке цикла $\alpha = 0$, то, в соответствии с (8), операция р-итерации завершится на этом витке, что противоречит условию теоремы.

Поскольку при всех сделанных предположениях операция вырождается, то сделанные предположения неверны, а заданные условия являются необходимыми для завершения р-итерации.

Теорема доказана.

Из доказанной теоремы следует, что на выходе Д-оператора $(D)O^{u}(D')$ в качестве подмножества дополнительных данных специфицируется специальный тип связывающих данных $\vec{D}_i^{CB} = D_i \cap D_i'$.

Возможность спецификации связывающих данных в КС проиллюстрируем записью КС (4) в следующем виде:

$$\begin{aligned}(D^{ИСХ})A(D^{РЕЗ}) = &({}_1D_1^{ИСХ}, {}_1D_1'')\,{}_1P_1({}_1D_1^{РЕЗ}, {}_1D_1''', {}_1\vec{D}_1^{CB})*\ldots\\ \ldots*&({}_1\vec{D}_i^{CB}, {}_1D_i^{ИСХ}, {}_1D_i'')\,{}_1P_i({}_1D_i^{РЕЗ}, {}_1D_i''', {}_1\vec{D}_i^{CB})*\ldots\\ \ldots*&({}_1\vec{D}_k^{CB}, {}_1D_k^{ИСХ}, {}_1D_k'')\,{}_1P_k({}_1D_k^{РЕЗ}),\end{aligned} \tag{9}$$

где ${}_1D_i'' = {}_1D_i \setminus {}_1\vec{D}_k^{CB}$, ${}_1D_i'' = {}_1D_i' \setminus {}_1\vec{D}_k^{CB}$.

Из определения 6 следует, что в общем случае каждая подсистема, входящая в РС, может быть связана со всеми следующими за ней и всеми предшествующими ей подсистемами. Обычно не все операторы в КС связаны друг с другом, а, в соответствии с определением 6, множества ${}_j\vec{D}_p$ (${}_j\vec{D}_p$) могут быть пустыми. Более того, при $\vec{D}_i^{CB} = \varnothing$ и/или $\vec{D}_i^{CB} = \varnothing$ Д-оператор не имеет левых и/или правых связей.

Проиллюстрируем общий случай, когда в построенной КС все подсистемы связаны, а специфицированы не только связывающие данные, но и все "источники" и "приемники" данных поставлены в однозначное соответствие. Для этого перепишем КС (9) в виде:

$$\begin{aligned}(D^{ИСХ})A(D^{РЕЗ}) = &({}_1D_1^{ИСХ}, {}_1D_1'')\,{}_1P_1({}_1D_1^{РЕЗ}, {}_1D_1''', {}_1\vec{D}_2^1, {}_1\vec{D}_3^1, \ldots, {}_1\vec{D}_j^1, \ldots, {}_1\vec{D}_n^1)*\ldots\\ \ldots*&({}_1\vec{D}_i^1, {}_2\vec{D}_i^1, \ldots, {}_{i-1}\vec{D}_i^1, {}_1D_i^{ИСХ}, {}_1D_i'')\,{}_1P_i({}_1D_i^{РЕЗ}, {}_1D_i''', {}_i\vec{D}_{i+1}^1, {}_i\vec{D}_{i+2}^1, \ldots, {}_i\vec{D}_k^1)*\ldots\\ \ldots*&({}_1\vec{D}_k^1, {}_2\vec{D}_k^1, \ldots, {}_j\vec{D}_k^1, \ldots, {}_{k-1}\vec{D}_k^1, {}_1D_k^{ИСХ}, {}_1D_k'')\,{}_1P_k({}_1D_k^{РЕЗ}),\end{aligned} \tag{10}$$

где правый и левый нижние индексы в случае ${}_{i-1}\vec{D}_i^1$ и ${}_i\vec{D}_{i+1}^1$ означают номера приемников и источников данных (в соответствии со стрелкой), а правый верхний индекс – шаг разработки.

Из КС видно, что любому ${}_l\vec{D}_p^1$ соответствует ${}_l\vec{D}_p^1$, и, таким образом, выполняется соотношение $\bigcup_{\substack{s=2\\p=1}}^{\substack{k-1\\k}} {}_p\vec{D}_s^1 = \bigcup_{\substack{p=1\\s=2}}^{\substack{k-1\\k}} {}_s\vec{D}_p^1$.

Исключив верхние индексы, запишем КС (10), исходя из утверждения 1, для случая, когда каждая подсистема взаимодействует с ВУ:

$$(D^{ИСХ})A(D^{РЕЗ}) = ({}_1D_1^{ИСХ}, {}_1D_1^{исх}, {}_1D_1^{r\_u\_u}, {}_1D_1^{r}) {}_1P_1({}_1D_1^{РЕЗ}, {}_1D_1^{w\_p\_p}, {}_1D_1^{w}, {}_1\vec{D}_2^1, {}_1\vec{D}_3^1, \ldots$$
$$\ldots, {}_1\vec{D}_j^1, \ldots, {}_1\vec{D}_n^1) * \ldots$$
$$\ldots * ({}_1\vec{D}_i^1, {}_2\vec{D}_i^1, \ldots, {}_{i-1}\vec{D}_i^1, {}_1D_i^{ИСХ}, {}_1D_i^{исх}, {}_1D_i^{r\_u\_u}, {}_1D_i^{r}) P_i({}_1D_i^{РЕЗ}, {}_1D_i^{w\_p\_p}{}_i, {}_i\vec{D}_{i+1}^1, {}_i\vec{D}_{i+2}^1, \ldots$$
$$\ldots, {}_i\vec{D}_k^1) * \ldots$$
$$\ldots * ({}_1\vec{D}_k^1, {}_2\vec{D}_k^1, \ldots, {}_j\vec{D}_k^1, \ldots, {}_{k-1}\vec{D}_k^1, {}_1D_k^{ИСХ}, {}_1D_k^{исх}, {}_1D_k^{r\_u\_u}, {}_1D_1^{r}) {}_1P_k(D_k^{РЕЗ}, {}_1D_k^{w\_p\_p}).$$

По поводу вводимых и выводимых данных будем утверждать следующее.

Утверждение 4. Вводу исходных данных $D_i^{R\_исх}$ не может предшествовать их вывод, и они могут переходить в разряд вводимых. Вводимые и выводимые данные D_i^R, D_j^W (за исключением результирующих) являются связывающими ${}_i\vec{D}_j$ и реализуют частный случай связей в КС.

Доказательство. Поскольку, в соответствии с определением 3, данные $D_i^{R\_исх}$ имеют начальные значения, то вывод данных на их место приведет к утрате этих значений. После ввода $D_i^{R\_исх}$, если начальные значения более не требуются, то они могут использоваться для вывода, приобретая все свойства вводимых и выводимых данных. Так как вводимые копируются в память, а выводимые из памяти на ВУ, то, в соответствии с определением 6, эти данные являются связывающими.

Утверждение доказано.

Второй шаг разработки выполняется путем декомпозиции каждой подсистемы после того, как все данные на входе и выходе подсистем трактуются как глобальные. В результате чего получаем совокупность КС, каждый Д-оператор, входящий в которые, дополняется локальными данными, а КС записываются в виде:

$$({}_1D_1) {}_1P_1({}_1D_1') = ({}_2D_1) {}_2O_1({}_2D_1) * \ldots * ({}_2D_{n_1}) {}_2O_{n_1}({}_2D_{n_1})$$
$$\ldots$$
$$({}_1D_i) {}_1P_i({}_1D_i') = ({}_2D_{n_{i-1}+1}) {}_2O_{n_{i-1}+1}({}_2D_{n_{i-1}+1}) * \ldots * ({}_2D_{n_i}) {}_2O_{n_i}({}_2D_{n_i})$$
$$\ldots$$
$$({}_1D_k) {}_1P_k({}_1D_k') = ({}_2D_{n_{k-1}+1}) {}_2O_{n_{k-1}+1}({}_2D_{n_{i-1}+1}) * \ldots * ({}_2D_{n_k}) {}_2O_{n_k}({}_2D_{n_k})$$

и образуют первый “слой” алгоритма.

Поскольку данные, специфицированные на входе и выходе глобальные, а дополнительные данные имеют состав и структуру, задаваемую утверждением 1 и определениями 3, 6, то для них выполняются все доказанные свойства, а соотношения (5) записываются в виде: ${}_1D_i^{ИСХ} \subseteq \bigcup_{i=n_{i-1}+1}^{n_i} {}_2D_i^{ИСХ}$, ${}_1D_i^{РЕЗ} \subseteq \bigcup_{i=n_{i-1}+1}^{n_i} {}_2D_i^{РЕЗ}$.

Процесс проектирования продолжается аналогичным образом с получением все более детального описания алгоритма в виде новых слоев этого описания. Этот процесс продолжается до получения КС, образованных некоторыми элементарными Д-операторами. При этом свойства данных, доказанные в утверждениях 2–4, имеют место на всех этапах разработки.

В работе [21] показана возможность получения такого уровня детальности представления алгоритма, при котором возможен бесшовный переход от полученного таким образом описания алгоритма к программе. А в работе [22] – возможность автоматизации процесса проектирования.

Выводы из данного исследования и перспективы дальнейших исследований в данном направлении. В работе показан состав и структура данных, специфицированных на входах и выходах Д-операторов, входящих в композиционные схемы. В частности, введены исходные, результирующие, вводимые, выводимые и связывающие (образующие информационные связи в алгоритме) данные.

Продемонстрирована возможность описания информационных связей, существующих в этих схемах. При этом специфицированы не только связывающие данные, но и все "источники" и "приемники" данных поставлены в однозначное соответствие.

Показаны свойства данных в композиционных схемах, посредством которых описываются алгоритмы.

Полученные свойства данных и спецификации информационных связей в алгоритмах могут использоваться в качестве средств контроля его корректности с точки зрения обработки данных. И такой контроль может осуществляться на всех уровнях декомпозиции алгоритма.

Дальнейший анализ информационных связей и выявление новых свойств данных в алгоритмах является перспективным направлением будущих исследований для решения задач оценки информационной сложности алгоритмов и их оптимизирующих преобразований.

Перспективным направлением дальнейшего развития рассматриваемого алгебраического аппарата является формализация потоков данных в алгоритмах с целью распараллеливания последних.

Список использованных источников:

1. Данные в языках программирования: абстракция и типология : сб. статей / под ред. В. Агафонова. – М. : Мир, 1982. – 328 с.

2. Турский В. Методология программирования / Турский В. – М. : Мир, 1981. – 264 с.

3. Вирт Н. Алгоритмы + структуры данных = программы : пер. с англ. / Вирт Н. – М. : Мир, 1985. – 656 с.

4. Шнейдерман Б. Психология программирования: человеческие факторы в вычислительных и информационных системах / Шнейдерман Б. – М. : Радио и связь, 1984. – 304 с.

5. Bastani F. B. The effect of data structures on the logical complexity of programs / F. B. Bastani, S. S. Iyengar // CACM. – 1987. – Vol. 30. – № 3. – P. 250–259.

6. Глушков В. М. Алгебра. Языки. Программирование / Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л. – 3-е изд., перераб. и доп. – К. : Наук. думка, 1989. – 376 с.

7. Акуловский В. Г. Расширенная алгебра алгоритмов / В. Г. Акуловский // Проблемы программирования. – 2007. – № 3. – С. 3–15.

8. Акуловский В. Г. Алгебра алгоритмов, базирующаяся на данных / В. Г. Акуловский // Кибернетика и системный анализ. – 2012. – № 2. – С. 151–166.

9. Акуловский В. Г. Основы алгебры алгоритмов, базирующейся на данных / В. Г. Акуловский // Проблемы программирования. Спец. выпуск по материалам VII Международной научно-практической конференции по программированию УкрПРОГ 2010. – 2010. – № 2–3. – С. 89–96.

10. Akulovskiy V. G. Data based algorithmic algebra / V. G. Akulovskiy // Cybernetics and Systems Analysis. – 2012. – Vol. 48. – Issue 2. – P. 291–303.

11. Цейтлин Г. Е. Введение в алгоритмику / Цейтлин Г. Е. – К. : Сфера, 1998. – 310 с.

12. Акуловський В. Г. Деякі аспекти формалізації та специфікації інформаційних зв'язків в алгоритмах / В. Г. Акуловський, В. В. Костенко // Вісник Академії митної служби України. Серія: "Технічні науки". – 2009. – № 2. – С. 105–114.

13. Акуловський В. Г. Деякі властивості даних у композиційних схемах алгоритмів / В. Г. Акуловський, В. В. Костенко, В. В. Поліщук // Вісник Академії митної служби України. Серія: "Технічні науки". – 2010. – № 1. – С. 92–101.

14. Акуловский В. Г. Состав и свойства данных, специфицируемых в композиционных схемах алгоритмов / В. Г. Акуловский // Проблемы программирования. – 2013. – № 2. – С. 3–12.

15. Акуловский В. Г. Формализация взаимосвязей операторов и данных в рамках расширенной алгебры алгоритмов / В. Г. Акуловский // Кибернетика и системный анализ. – 2008. – № 6. – С. 170–182.

16. Akulovsky V. G. Formalization of interrelations between operators and data within the framework of an extended algebra of algorithms / V. G. Akulovsky // Cybernetics and Systems Analysis. – 2008. – Vol. 44. – Issue 6. – P. 941–950.

17. Дорошенко А. Ю. Висхідне проектування алгоритмів при алгеброалгоритмічному підході / А. Ю. Дорошенко, В. Г. Акуловський // Вісник Київського національного університету імені Тараса Шевченка. Серія: "Фізико-математичні науки". – 2012. – Вип. 1. – С. 167–172.

18. Дорошенко А. Е. Нисходящее проектирование алгоритмов в рамках алгеброалгоритмического подхода / А. Е. Дорошенко, В. Г. Акуловский // Математичні машини і системи. – 2012. – № 3. – С. 97–102.

19. Дорошенко А. Е. Алгоритмическое описание взаимодействия алгоритмов с внешними устройствами / А. Е. Дорошенко, В. Г. Акуловский // Управляющие системы и машины. – 2012. – № 3. – С. 45–53.

20. Акуловский В. Г. Реализация комплексного подхода к описанию алгоритмов информационно-управляющих систем в рамках алгебраического аппарата / В. Г. Акуловский, А. Е. Дорошенко // Управляющие системы и машины. – 2013. – № 5. – С. 46–52.

21. Акуловский В. Г. Реализация формализованного перехода от алгоритма к программе средствами расширенной алгебры алгоритмов / В. Г. Акуловский // Проблемы программирования. – 2008. – № 1. – С. 51–59.

22. Акуловский В. Г. Реализация средств проектирования и генерации программ на основе алгебры алгоритмов с данными / В. Г. Акуловский, А. Е. Дорошенко, Е. А. Яценко // Проблемы программирования. – 2015. – № 2. – С. 41–51.