УДК 519.683

В.Г. АКУЛОВСКИЙ, А.Е. ДОРОШЕНКО

ПРЕОБРАЗОВАНИЕ АЛГОРИТМОВ, ЗАПИСАННЫХ В ВИДЕ КОМПОЗИЦИОННЫХ СХЕМ

Аннотация. Рассмотрены свойства данных, специфицируемых в композиционных схемах алгоритмов, в частности, свойства данных, образующих информационные связи между Д-операторами, входящими в композиционные схемы. Показана возможность преобразования как композиционных схем, описывающих фрагменты алгоритма, так и всего алгоритма путем перемещения Д-операторов в них.

Ключевые слова: алгоритмы, алгебра алгоритмов с данными, композиционные схемы алгоритмов, преобразование алгоритмов.

ВВЕДЕНИЕ

Исходя из важности роли, которую данные играют в программировании [1–3], авторы модифицировали известную модель ЭВМ [4]. Модификация заключается в дополнении модели внешней средой (ВС), представляющей собой, с одной стороны, память и внешние устройства (ВУ), а с другой — множество данных $D^{\text{ВС}}$.

Кроме того, в рамках выполненной модификации введены Д-операторы — операторы вида $(D)O(D')$ (этим и обусловлено используемое название) со специфицированными на входе входными или обрабатываемыми и на выходе выходными или продуцируемыми данными такими, что $D \subseteq D^{\text{ВС}}$ и $D' \subseteq D^{\text{ВС}}$. Введем следующее определение [5].

Определение 1. Данными называется пара $D = \langle N, З \rangle$, где N — их носитель, З — кортеж значений, носителем которых является N. На каждом шаге вычислительного процесса носитель содержит (хранит) некоторый (текущий) кортеж значений данных, в частности, эти значения могут быть неопределенными.

Функционирование модифицированной модели ЭВМ обеспечивает вычислительный процесс, в ходе которого в результате выполнения каждого Д-оператора обрабатываются, т.е. анализируются и преобразуются специфицированные данные. В результате выполненных преобразований в общем случае изменяются значения как обрабатываемых (входных), так и продуцируемых (выходных) данных, при неизменном составе носителей этих значений (см. определение 1). Отсюда следует, что специфицируются фактически носители данных. В связи с этим под теоретико-множественными операциями, определенными на множествах данных, будем понимать операции над их носителями. В частности, под соотношением $D = D'$ будем подразумевать совпадение множеств носителей данных D, D'. В том случае, когда будем говорить о равенстве данных, понимая под этим равенство как множества носителей, так и множества значений, будем использовать символ «≡» ($D \equiv D'$), при совпадении множеств носителей при несовпадении множеств значений — символ «≢» ($D \not\equiv D'$).

На основе модифицированной модели ЭВМ предложена система алгоритмических алгебр (САА/Д) [6, 7], представляющая собой трехосновную алгебраическую систему $\langle U, L, D, \Omega \rangle$, основами которой являются множество Д-операторов U, множество логических условий L и множество данных D, а $\Omega = \Omega_1 \cup \Omega_2 \cup \Omega_3$ — ее сигнатура, состоящая из множества Ω_1 — операций, принимающих значения

на множестве U, Ω_2 — логических операций, принимающих значения на множестве L, и Ω_3 — операций, принимающих значения на множестве данных D.

Д-операторы служат для описания алгоритмов, а одной из форм такого описания являются композиционные схемы (КС) [6], в которых используется операция композиции (*) Д-операторов $(D_1)O_1(D_1')*(D_2)O_2(D_1')$, означающая последовательное выполнение сначала Д-оператора $(D_1)O_1(D_1')$, затем Д-оператора $(D_2)O_2(D_2')$.

С помощью КС реализуется две стратегии проектирования алгоритмов: нисходящая и восходящая [8, 9]. В первом, более распространенном и рассматриваемом в данной работе случае, исходный Д-оператор $(D)O(D')$ декомпозируется, а специфицированные на его входе и выходе данные детализируются, в результате его можно представить в виде КС:

$$(D)O(D') = (D_1)O_1(D_1')*(D_2)O_2(D_2')*...*(D_k)O_k(D_k'), \qquad (1)$$

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, $D_1,...,D_k$ — обрабатываемые данные такие, что $D \subseteq (D_1 \cup ... \cup D_k)$, а $D_1',...,D_k'$ — данные, полученные в результате обработки, такие, что $D' \subseteq (D_1' \cup ... \cup D_k')$.

Процесс декомпозиции осуществляется таким образом, чтобы обеспечить требуемую последовательность обработки данных, которая имеет следующую специфику. Производные Д-операторы обрабатывают данные как «самостоятельно», так и «коллективно», т.е. в общем случае каждый из производных Д-операторов, с одной стороны, обрабатывает данные независимо от других производных Д-операторов, с другой, обработка множества данных, начатая некоторым Д-оператором $(D_i)O_i(D_i')$, завершается и\или продолжается в результате выполнения другого Д-оператора $(D_j)O_j(D_j')$, где $j > i$. Таким образом, в последнем случае обработка результатов выполнения $(D_i)O_i(D_i')$ завершается выполнением Д-оператора $(D_j)O_j(D_j')$ или результаты, полученные после выполнения $(D_j)O_j(D_j')$, подвергаются дальнейшей обработке одним или несколькими следующими Д-операторами, которая завершается после выполнения некоторого $(D_p)O_p(D_p')$ такого, что $p > j$.

Кроме того, при обработке данных могут использоваться и, как правило, используются вспомогательные (локальные) данные, которые не специфицированы на входе исходного Д-оператора.

Предлагаемый алгебраический аппарат предназначен для описания произвольных алгоритмов, а одним из важнейших преимуществ такого аппарата является возможность формального преобразования этих алгоритмов (схем алгоритмов). Реализации возможности описания и преобразования алгоритмов в рамках предложенного алгебраического аппарата и посвящена данная работа. Единственным, по-видимому, механизмом преобразования алгоритмов в случае представления его в виде КС является перемещение Д-операторов. При этом специфика представления алгоритма требует для обеспечения адекватности таких преобразований учета свойств данных и особенностей их обработки. Решение задачи о перемещении Д-операторов в алгоритмах начнем с рассмотрения возможности их перемещения в рамках КС.

ПЕРЕМЕЩЕНИЕ Д-ОПЕРАТОРОВ В РАМКАХ КС

Определим некоторые виды специфицируемых данных. Поскольку, как отмечалось во Введении, носителями данных могут быть внешние устройства, определим вводимые и выводимые данные.

Определение 2. Множества данных, носителями которых являются некоторые ВУ, с которых эти данные считываются (копируются с ВУ), будем обозначать D^R и называть вводимыми, а на которые записываются (копируются на ВУ) D^W, — выводимыми. Вводимые и выводимые данные такие, что у произвольного Д-оператора $(D)O(D')$ $D^R \subseteq D$, $D^W \subseteq D'$ и могут быть пустыми ($DD^R = \varnothing$ и\или $D^W = \varnothing$), т.е. вводимые и\или выводимые данные могут отсутствовать. В КС (1) $D^R \subseteq (D_1^R \cup \ldots \cup D_k^R)$ и $D^W \subseteq (D_1^W \cup \ldots \cup D_k^W)$, где $D_i^R \subseteq D_i$ и $D_i^W \subseteq D_i'$.

Заметим, что Д-операторами, входящими в КС, в общем случае вводятся и выводятся локальные данные, т.е. данные, не специфицированные на входе и выходе исходного Д-оператора. Для решаемой задачи это несущественно.

Учитывая, что обработка множества данных, продуцируемых одним Д-оператором, может быть продолжена следующим (следующими) за ним Д-оператором, определим понятие связи между ними.

Определение 3. В КС Д-операторы $(D_i)O_i(D_i')$ и $(D_j)O_j(D_j')$ $(j > i)$ назовем информационно связанными (в дальнейшем связанными), если для специфицированных у них данных выполняется соотношение $D_i' \cap D_j \neq \varnothing$, в противном случае эти Д-операторы не связаны. Данные ${}_iD_j = (D_i' \cap D_j)$ назовем связывающими, если при обозначении их на выходе *i*-го Д-оператора ${}_i\vec{D}_j$, а на входе *j*-го — ${}_i\overleftarrow{D}_j$, выполняется соотношение ${}_i\vec{D}_j \equiv {}_i\overleftarrow{D}_j$. Если это соотношение не выполняется, то ${}_iD_j$ не являются связывающими данными. Данные, связывающие Д-оператор $(D_i)O_i(D_i')$ со всеми предшествующими ${}_1D_i^{\text{СВ}} = {}_1D_i \cup \ldots \cup {}_{i-1}D_i$ и всеми последующими ${}_iD_k^{\text{СВ}} = {}_iD_{i+1} \cup \ldots \cup {}_iD_k$ Д-операторами, назовем связывающими Д-оператор $(D_i)O_i(D_i')$ соответственно слева и справа. Определенные связи являются локальными — инкапсулированными в КС.

Определение 3, естественно, распространяется и на случай $(D_i)O_i(D_i')*$ $*(D_{i+1})O_{i+1}(D_{i+1}')$, где ${}_iD_{i+1}$ — данные, связывающие эти Д-операторы.

По поводу связывающих данных будем утверждать следующее.

Утверждение 1. На входе первого $(D_1)O_1(D_1')$ и последнего $(D_k)O_k(D_k')$ в КС Д-операторов спецификации локальных связывающих данных отсутствуют.

Доказательство очевидно и вытекает из определения 3, так как для первого Д-оператора отсутствуют предшествующие, а для последнего — последующие Д-операторы.

Процесс перемещения Д-операторов будем рассматривать как результат последовательной перестановки соседних Д-операторов. Для этого остановимся на факторах, ограничивающих возможности такой перестановки в случае $(D_i)O_i(D_i')*(D_{i+1})O_{i+1}(D_{i+1}')$.

Первым таким фактором являются операции ввода-вывода, которые, очевидно, выполняют Д-операторы, на входе и\или выходе которых специфицированы вводимые D^R и\или выводимые D^W данные. Последовательность операций ввода–вывода в конкретном алгоритме весьма часто не является строго заданной, т.е. в общем случае для него существует множество корректных (допустимых) последовательностей, любая из которых не нарушает выполнение этого алгоритма. В связи с этим будем полагать, что в некоторых случаях корректность последовательности операций ввода–вывода будет сохраняться при изменении порядка следования Д-операторов, заданного КС. Кроме того, будем считать, что существует возможность проверки такой корректности, которая в связи с отсутствием формализации реализуется «вручную».

Вторым фактором являются информационные связи, которые определяют неизменный порядок обработки данных, так как при их наличии Д-оператор $(D_{i+1})O_{i+1}(D'_{i+1})$ продолжает обработку данных, начатую Д-оператором $(D_i)O_i(D'_i)$, и, таким образом, поменять их местами невозможно.

Для того чтобы в дальнейшем учесть третий фактор, введем следующее определение.

Определение 4. Д-операторы $(D_i)O_i(D'_i)$ и $(D_{i+1})O_{i+1}(D'_{i+1})$ имеют между собой обратные связи, если выполняется соотношение $D_i \cap D'_{i+1} \neq \varnothing$, выполнение которого изменяет значения данных, специфицированных на входе $(D_i)O_i(D'_i)$, и соотношение $D'_i \cap D'_{i+1} \neq \varnothing$, при выполнении которого изменяются значения данных, продуцируемых Д-оператором $(D_i)O_i(D'_i)$.

Определим условие, при котором возможна перестановка Д-операторов и докажем возможность их перемещения в КС.

Определение 5. Первым условием адекватности преобразованной КС, в которой композиция $(D_{i+1})O_{i+1}(D'_{i+1})*(D_i)O_i(D'_i)$ получена в результате перестановки входящих в нее Д-операторов, является то, что значения всех данных, полученных в результате выполнения исходной композиции, должны полностью совпадать со значениями тех же данных, полученных после выполнения производной композиции Д-операторов. Вторым условием является допустимая последовательность операций ввода–вывода.

Теорема 1. В КС любой Д-оператор $(D_i)O_i(D'_i)$ может перемещаться вправо (влево) в результате последовательной перестановки его с соседними справа (слева) Д-операторами при условии, что при каждой такой перестановке данный Д-оператор не связан $D'_i \cap D_{i+1} = \varnothing$ и не имеет обратных связей $D_i \cap D'_{i+1} = \varnothing$, $D'_i \cap D'_{i+1} = \varnothing$ с соседним, а последовательность операций ввода–вывода, если вводимые и выводимые данные специфицированы на входах и выходах этих Д-операторов, остается допустимой.

Доказательство начнем с рассмотрения случая перемещения Д-оператора $(D_i)O_i(D'_i)$ вправо. Результатами выполнения композиции Д-операторов $(D_i)O_i(D'_i)*(D_{i+1})O_{i+1}(D'_{i+1})$ будет множество данных D'_{i+1}, значения которых являются результатом выполнения второго $(i+1)$-го, и только второго Д-оператора, что очевидно. И множество данных D'_i, значения которых являются результатом выполнения первого (i)-го и только первого Д-оператора. Последний результат обусловлен отсутствием между Д-операторами, по условию теоремы, связи $D'_i \cap D_{i+1} = \varnothing$ и обратной связи $D'_i \cap D'_{i+1} = \varnothing$. Поскольку в соответствии с определениями 3 и 4 данные, продуцируемые первым Д-оператором, дальнейшей обработке не подвергаются, а результат D'_{i+1}, продуцируемый вторым Д-оператором, не изменяет значений данных D'_i.

В результате перестановки Д-операторов будет получена композиция $(D_{i+1})O_{i+1}(D'_{i+1})*(D_i)O_i(D'_i)$, итогом выполнения которой будут множество данных D'_i, значения которых, очевидно, являются результатом выполнения второго (i)-го, и только второго Д-оператора, и множество данных D'_{i+1}, значения которых являются результатом выполнения первого $(i+1)$-го, и только первого Д-оператора. Последний результат обусловлен отсутствием связи и обратной связи между Д-операторами, так как по условию теоремы $D_i \cap D'_{i+1} = \varnothing$ и $D'_i \cap D'_{i+1} = \varnothing$. В этом случае в соответствии с определениями 3 и 4 значения D'_{i+1}, продуцируемые Д-оператором $(D_{i+1})O_{i+1}(D'_{i+1})$, дальнейшей обработке Д-оператором $(D_i)O_i(D'_i)$ не подвергаются, а результат выполнения этого Д-оператора D'_i на полученные значения D'_{i+1} не влияет.

В обоих случаях значения всех получаемых данных полностью совпадают и, таким образом, выполняется первое условие, заданное определением 5. Если вво-

димые и выводимые данные у переставляемых Д-операторов не специфицированы или специфицированы только у одного из них, то выполнение этого условия оказывается достаточным. В противном случае, если в результате проверки окажется, что перестановка Д-операторов не нарушает порядок выполнения операций ввода–вывода, то в соответствии с определением 5 полученная композиция Д-операторов адекватна исходной, т.е. композиция Д-операторов $(D_i)O_i(D'_i)*(D_{i+1})O_{i+1}(D'_{i+1})$ преобразована в композицию $(D_{i+1})O_{i+1}(D'_{i+1})*(D_i)O_i(D'_i)$.

Этот процесс, очевидно, можно продолжить для случая $(D_i)O_i(D'_i)*$ $*(D_{i+2})O_{i+2}(D'_{i+2})$ и т.д. при соблюдении условий, сформулированных в теореме.

Таким образом, Д-оператор может быть перемещен в КС вправо. Доказательство для случая перемещения влево полностью аналогично.

Теорема доказана.

Следствие 1. Полученный результат может быть обобщен для случая последовательности Д-операторов $(D_j)O_j(D'_j)*(D_{j+1})O_{j+1}(D'_{j+1})*\ldots*(D_p)O_p(D'_p)$, поскольку в работе [9] показана возможность объединения Д-операторов. Если в результате объединения будет получен Д-оператор $D_i = (D_j)O_j(D'_j)*\ldots$ $\ldots*(D_p)O_p(D'_p)$, который удовлетворяет условиям теоремы, то указанную последовательность Д-операторов можно перемещать в рамках КС.

Следствие 2. Перемещение Д-операторов и их последовательностей в КС ограничено заданными условиями и в общем случае такое перемещение возможно в некоторой области, которую назовем областью мобильности Д-оператора.

Показав возможность перемещения Д-операторов в КС, перейдем к рассмотрению алгоритмов.

ОПИСАНИЕ И ПРЕОБРАЗОВАНИЕ АЛГОРИТМОВ

Алгоритм определим следующим образом, при этом данные детализируем с учетом определения 2.

Определение 6. Д-оператор $(D, D^R)A(D^W)$, на входе которого специфицированы все необходимые для решения некоторой задачи глобальные обрабатываемые данные, а на выходе — все требуемые глобальные результирующие данные, будем называть исходным (нулевым) уровнем представления алгоритма решения упомянутой задачи или просто алгоритмом.

Для того чтобы выполнить первый этап разработки, т.е. декомпозировать алгоритм и перейти к следующему уровню его представления, определим производный Д-оператор, в рамках которого специфицируемые данные также детализируются.

Определение 7. Производный Д-оператор $(D_i^R, D_i^G, D_i, {}_1\vec{D}_i^{\text{СВ}})O_i(D_i^W, {}_i\vec{D}_k^{\text{СВ}})$, входящий в КС, обрабатывает подмножество вводимых данных D_i^R, подмножество глобальных данных $D_i^G \subseteq D$, для обработки которых используются локальные данные D_i. В результате обработки продуцируются выводимые данные D_i^W. Множество ${}_i\vec{D}_k^{\text{СВ}}$, образованное всеми левыми связями этого Д-оператора, — это данные, обработка которых начата предыдущими Д-операторами и им продолжается. Множество ${}_i\vec{D}_k^{\text{СВ}}$, образованное всеми правыми связями этого Д-оператора, — это данные, обработка которых будет продолжена следующими Д-операторами.

Первый этап разработки алгоритма будем называть архитектурным, а компоненты полученной КС-подсистемами (подзадачами), которые реализуют все функции по преобразованию всех исходных данных во все результирующие, т.е. все глобальные обрабатываемые и результирующие данные распределяются

между подсистемами. Рассмотрим общий случай, когда эти данные распределяются между всеми подсистемами, а их обработка, начатая одной подсистемой, продолжается другими.

Отметим, что в частных простых случаях этот этап может отсутствовать, а в сложных реализоваться за несколько шагов.

Исходя из определений 6, 7 и соотношения 1, выполним первый этап декомпозиции алгоритма, записав КС в виде

$$(D, D^R)A(D^W) = (D_1^R, D_1^G, D_1)O_1(D_1^W, {}_1\vec{D}_k^{\mathrm{CB}})*\ldots$$

$$\ldots*(D_i^R, D_i^G, D_i, {}_1\vec{D}_i^{\mathrm{CB}})O_i(D_i^W, {}_i\vec{D}_k^{\mathrm{CB}})*\ldots$$

$$\ldots*(D_k^R, D_k^G, D_k, {}_1\vec{D}_k^{\mathrm{CB}})O_k(D_k^W),$$

где в соответствии с утверждением 1 первый Д-оператор на входе и последний на выходе связывающих данных не имеют, $D = (D_1^G \cup \ldots \cup D_k^G)$, $D^R \subseteq (D_1^R \cup \ldots \ldots \cup D_k^R)$, $D^W \subseteq (D_1^W \cup \ldots \cup D_k^W)$.

На втором этапе декомпозиции алгоритма, где декомпозируется каждый производный на предыдущем уровне Д-оператор, получим некоторую совокупность КС, которую назовем слоем алгоритма.

Запишем КС, образующие первый слой алгоритма, при записи которых с помощью верхнего левого индекса будем указывать номер слоя, а для производных Д-операторов использовать сквозную нумерацию:

$$(D_1^R, D_1^G, D_1)O_1(D_1^W, {}_1\vec{D}_k^{\mathrm{CB}}) = ({}^1D_1^R, {}^1D_1^G, {}^1D_1)\,{}^1O_1({}^1D_1^W, {}_1^1\vec{D}_{1k}^{\mathrm{CB}})*\ldots$$

$$\ldots*({}^1D_{k_1}^R, {}^1D_{k_1}^G, {}^1D_{k_1}, {}_1^1\vec{D}_{k_1}^{\mathrm{CB}})\,{}^1O_{k_1}({}^1D_{k_1}^W, {}_{k_1}^1\vec{D}_{1k}^{\mathrm{CB}}),$$

$$\ldots$$

$$(D_i^R, D_i^G, D_i, {}_1\vec{D}_i^{\mathrm{CB}})O_i(D_i^W, {}_i\vec{D}_k^{\mathrm{CB}}) =$$

$$= ({}^1D_{k_{i-1}+1}^R, {}^1D_{k_{i-1}+1}^G, {}^1D_{k_{i-1}+1}, {}_1^1\vec{D}_{k_{i-1}+1}^{\mathrm{CB}})\,{}^1O_{k_{i-1}+1}({}^1D_{k_{i-1}+1}^W, {}_{k_{i-1}+1}^1\vec{D}_{1k}^{\mathrm{CB}})*\ldots$$

$$\ldots*({}^1D_{k_i}^R, {}^1D_{k_i}^G, {}^1D_{k_i}, {}_1^1\vec{D}_{k_i}^{\mathrm{CB}})O_{k_i}(D_{k_i}^W, {}_{k_i}^1\vec{D}_{1k}^{\mathrm{CB}}),$$

$$\ldots$$

$$(D_k^R, D_k^G, D_k, {}_1\vec{D}_k^{\mathrm{CB}})O_k(D_k^W) =$$

$$= ({}^1D_{k_{k-1}+1}^R, {}^1D_{k_{k-1}+1}^G, {}^1D_{k_{k-1}+1}, {}_1^1\vec{D}_{k_{k-1}+1}^{\mathrm{CB}})\,{}^1O_{k_{k-1}+1}({}^1D_{k_{k-1}+1}^W, {}_{k_{k-1}+1}^1\vec{D}_{1k}^{\mathrm{CB}})*\ldots$$

$$\ldots*({}^1D_{{}_1^1k}^R, {}^1D_{{}_1^1k}^G, {}^1D_{{}_1^1k}, {}_1^1\vec{D}_{{}_1^1k}^{\mathrm{CB}})\,{}^1O_{{}_1^1k}({}^1D_{{}_1^1k}^W),$$

где ${}^1k$ — номер последнего Д-оператора в слое.

В каждой КС, образующей слой, глобальными являются данные исходного Д-оператора, включая и локальные D_i, введенные на предыдущем этапе описания.

При этом связывающие данные ${}_1^1\vec{D}_{k_j}^{\mathrm{CB}}$ и ${}_i^1\vec{D}_{k_j}^{\mathrm{CB}}$ включают как локальные, появившиеся в результате декомпозиции, так и глобальные данные, перешедшие с предыдущего уровня представления алгоритма. За исключением первого и последнего Д-операторов в слое, на входе и выходе которых связывающие данные отсутствуют, и первого и последнего Д-операторов в каждой КС, на входе и выходе которых специфицированы только глобальные данные, что следует из определения 7 и утверждения 1.

Таким образом, глобальные связывающие данные связывают Д-операторы из различных КС в рамках одного слоя.

В результате очередной декомпозиции Д-операторов получаем следующий слой, отличный от предыдущего большей «толщиной». Этот процесс продолжается до достижения требуемого уровня детализации описания алгоритма.

Обобщая проведенные рассуждения, запишем j-й слой алгоритма:

$$({}^{j-1}D_1^R, {}^{j-1}D_1^G, {}^{j-1}D_1)\,{}^{j-1}O_1({}^{j-1}D_1^W, {}_1^{j-1}\vec{D}^{\mathrm{CB}}_{{}^{j-1}k}) =$$

$$= ({}^{j}D_1^R, {}^{j}D_1^G, {}^{j}D_1)\,{}^{j}O_1({}^{j}D_1^W, {}_1^{j}\vec{D}^{\mathrm{CB}}_{{}^{j}k})*\ldots$$

$$\ldots*({}^{j}D_{k_1}^R, {}^{j}D_{k_1}^G, {}^{j}D_{k_1}, {}_1^{j}\tilde{D}^{\mathrm{CB}}_{k_1})\,{}^{j}O_{k_1}({}^{j}D_{k_1}^W, {}_{k_1}^{j}\vec{D}^{\mathrm{CB}}_{{}^{j}k}),$$

$$\ldots$$

$$({}^{j-1}D_i^R, {}^{j-1}D_i^G, {}^{j-1}D_i, {}_1^{j-1}\tilde{D}_i^{\mathrm{CB}})\,{}^{j-1}O_i({}^{j-1}D_i^W, {}_1^{j-1}\vec{D}_k^{\mathrm{CB}}) =$$

$$= ({}^{j}D^R_{k_{i-1}+1}, {}^{j}D^G_{k_{i-1}+1}, {}^{j}D_{k_{i-1}+1}, {}_1^{j}\tilde{D}^{\mathrm{CB}}_{k_{i-1}+1})\,{}^{j}O_{k_{i-1}+1}({}^{j}D^W_{k_{i-1}+1}, {}^{j}_{k_{i-1}+1}\vec{D}^{\mathrm{CB}}_{{}^{j}k})*\ldots$$

$$\ldots*({}^{j}D_{k_i}^R, {}^{j}D_{k_i}^G, {}^{j}D_{k_i}, {}_1^{j}\tilde{D}^{\mathrm{CB}}_{k_i})\,{}^{j}O_{k_i}({}^{j}D_{k_i}^W, {}_{k_i}^{j}\vec{D}^{\mathrm{CB}}_{{}^{j}k}),$$

$$\ldots$$

$$({}^{j-1}D_k^R, {}^{j-1}D_k^G, D_k, {}_1^{j-1}\tilde{D}_i^{\mathrm{CB}})\,{}^{j-1}O_k({}^{j-1}D_k^W) =$$

$$= ({}^{j}D^R_{k_{k-1}+1}, {}^{j}D^G_{k_{k-1}+1}, {}^{j}D_{k_{k-1}+1}, {}_1^{j}\tilde{D}^{\mathrm{CB}}_{k_{k-1}+1})\,{}^{j}O_{k_{k-1}+1}({}^{j}D^W_{k_{k-1}+1}, {}^{j}_{k_{k-1}+1}\vec{D}^{\mathrm{CB}}_{{}^{j}k})*\ldots$$

$$\ldots*({}^{j}D_{{}^{j}k}^R, {}^{j}D_{{}^{j}k}^G, {}^{j}D_{{}^{j}k}, {}_1^{j}\tilde{D}^{\mathrm{CB}}_{{}^{j}k})\,{}^{j}O_{{}^{j}k}({}^{j}D^W_{{}^{j}k}),$$

где ${}^{j}k$ — номер последнего Д-оператора в j-м слое.

Таким образом, показана возможность формального описания произвольного алгоритма и процесс его последовательной декомпозиции и детализации обрабатываемых данных. Прокомментируем полученные результаты. На входе и выходе алгоритма специфицируются все глобальные данные, подлежащие обработке, и данные, представляющие собой результат этой обработки. На первом этапе эти данные детализуются и распределяются между подсистемами, которые реализуют все функции алгоритма по их обработке. При этом эти данные дополняются вспомогательными локальными данными, а все связи на этом этапе являются локальными.

При переходе ко второму и всем последующим этапам все данные и связи, имевшие место на предыдущем этапе, трактуются как глобальные, а локальные вводятся на текущем уровне декомпозиции.

Отметим, что все связи внутри слоя алгоритма как глобальные, так и локальные, в случае детализации связывающих данных могут быть специфицированы (см. [10]).

Получив описание алгоритма, перейдем к рассмотрению возможностей его преобразования, которое, как отмечалось выше, представляет собой перемещение Д-операторов, входящих в описание алгоритма. В связи с тем, что в процессе описания алгоритма специфицированные данные были детализированы, ослабим ограничения, налагаемые определениями 3 и 4, из которых следует, что Д-операторы не связаны при условии $(D_i^W \cup {}_i\vec{D}_k^{\mathrm{CB}}) \cap (D_{i+1}^R \cup D_{i+1}^G \cup D_{i+1} \cup {}_1\tilde{D}_{i+1}^{\mathrm{CB}}) = \varnothing$ и не имеют обратных связей при условиях

$$(D_i^R \cup D_i^G \cup D_i \cup {}_1\vec{D}_i^{\text{CB}}) \cap (D_{i+1}^W \cup {}_{i+1}\vec{D}_k^{\text{CB}}) = \varnothing,$$
$$(D_i^W \cup {}_i\vec{D}_k^{\text{CB}}) \cap (D_{i+1}^W \cup {}_{i+1}\vec{D}_k^{\text{CB}}) = \varnothing. \quad (2)$$

Для этого приведем следующие утверждения.

Утверждение 2. В композиции

$$(D_i^R, D_i^G, D_i, {}_1\vec{D}_i^{\text{CB}})O_i(D_i^W, {}_i\vec{D}_k^{\text{CB}})^*(D_{i+1}^R, D_{i+1}^G, D_{i+1}, {}_1\vec{D}_{i+1}^{\text{CB}})O_{i+1}(D_{i+1}^W, {}_{i+1}\vec{D}_k^{\text{CB}})$$

Д-операторы не связаны, если выполняется соотношение ${}_iD_{i+1} = \varnothing$.

Доказательство. Поскольку в соответствии с определением 3 данные ${}_iD_{i+1}$, связывающие Д-операторы, такие, что ${}_iD_{i+1} \subseteq {}_i\vec{D}_k^{\text{CB}}$, а ${}_i\vec{D}_k^{\text{CB}}$ — это все данные, связывающие *i*-й Д-оператор справа, то при ${}_iD_{i+1} = \varnothing$ эти Д-операторы не связаны.

Утверждение доказано

Утверждение 3. В композиции Д-операторов

$$(D_i^R, D_i^G, D_i, {}_1\vec{D}_i^{\text{CB}})O_i(D_i^W, {}_i\vec{D}_k^{\text{CB}})^*(D_{i+1}^R, D_{i+1}^G, D_{i+1}, {}_1\vec{D}_{i+1}^{\text{CB}})O_{i+1}(D_{i+1}^W, {}_{i+1}\vec{D}_k^{\text{CB}})$$

обратные связи отсутствуют, если выполняются соотношения $(D_i^R \cup D_i^G \cup \cup D_i \cup {}_1\vec{D}_i^{\text{CB}}) \cap {}_{i+1}D_k = \varnothing$ и ${}_i\vec{D}_k^{\text{CB}} \cap {}_{i+1}\vec{D}_k^{\text{CB}} = \varnothing$.

Доказательство. Поскольку в соответствии с определением 2 данные D_{i+1}^W копируются на ВУ, то выполнение соотношений $(D_i^R \cup D_i^G \cup D_i \cup \cup {}_1\vec{D}_i^{\text{CB}}) \cap D_{i+1}^W \neq \varnothing$ и $(D_i^W \cup {}_i\vec{D}_k^{\text{CB}}) \cap D_{i+1}^W \neq \varnothing$ (см. (2)) не приведет к изменению значений ни одного из множеств данных. Таким образом, выводимыми данными в этих соотношениях можно пренебречь, в результате получим соотношения, указанные в утверждении.

Утверждение доказано.

С учетом сделанных уточнений покажем возможность преобразования алгоритма.

Теорема 2. Любой Д-оператор и любая последовательность Д-операторов при соблюдении условий, сформулированных в теореме 1 и следствиях из нее, может перемещаться в рамках алгоритма в области их мобильности. При этом условия перестановки Д-операторов в процессе перемещения — отсутствие связей ${}_iD_{i+1} = \varnothing$ и обратных связей $(D_i^R \cup D_i^G \cup D_i \cup {}_1\vec{D}_i^{\text{CB}}) \cap {}_{i+1}D_k = \varnothing$ и ${}_i\vec{D}_k^{\text{CB}} \cap {}_{i+1}\vec{D}_k^{\text{CB}} = \varnothing$.

Доказательство. Осуществляя свертку описания алгоритма, которое состоит из *j* слоев, получаем в результате последовательной подстановки во всех слоях вместо производных Д-операторов исходных, единственную КС, описывающую алгоритм в виде

$$(D, D^R)A(D^W) = (D_1^R, D_1^G, D_1)O_1(D_1^W, {}_1\vec{D}_{j_k}^{\text{CB}})^*\ldots$$
$$\ldots{}^*(D_i^R, D_i^G, D_i, {}_1\vec{D}_i^{\text{CB}})O_{k_1}(D_{k_1}^W, {}_i\vec{D}_{j_k}^{\text{CB}})^*\ldots$$
$$\ldots{}^*(D_{j_k}^R, D_{j_k}^G, D_{j_k}, {}_1\vec{D}_{j_k}^{\text{CB}})O_{j_k}(D_{j_k}^W),$$

в которой образующие ее Д-операторы входили в последний (заключительный) слой алгоритма, а ${}^jk$ — номер последнего Д-оператора в построенной КС.

В результате на нее распространяется результат, полученный в теореме 1 и следствиях 1, 2 из нее, т.е. любой Д-оператор может перемещать в рамках алгоритма в пределах его зоны мобильности. При этом условия перестановки Д-опе-

раторов в процессе перемещения — отсутствие связей ${}_iD_{i+1} = \varnothing$ и обратных связей $(D_i^R \cup D_i^G \cup D_i \cup {}_1\bar{D}_i^{\text{СВ}}) \cap {}_{i+1}\vec{D}_k^{\text{СВ}} = \varnothing$ и ${}_i\vec{D}_k^{\text{СВ}} \cap {}_{i+1}\vec{D}_k^{\text{СВ}} = \varnothing$ определены утверждениями 2 и 3.

Теорема доказана.

ЗАКЛЮЧЕНИЕ

В результате выполненной работы показана возможность в рамках САА\Д не только осуществлять формальное поэтапное (поуровневое) описание произвольных алгоритмов, но и преобразовывать описанные алгоритмы перемещением входящих в описание алгоритмов Д-операторов. Заметим, что возможность перемещения Д-операторов не зависит от стратегии проектирования алгоритмов.

Интуитивно понятно, что такие преобразования позволяют принципиально изменять (улучшать, оптимизировать) как свойства, так и характеристики алгоритмов. Хотя формально оценить все результаты таких преобразований на данном этапе затруднительно, по крайней мере, длина информационных связей в КС может быть уменьшена, что показано в работе [11]. Из настоящей публикации следует, что такая возможность распространяется на весь алгоритм.

Выполнение формальных оптимизирующих преобразований алгоритмов путем перемещения Д-операторов в них является направлением дальнейших исследований. Кроме того, распространение полученных результатов на случаи представления алгоритмов в виде композиционно-регулярных и регулярных схем — необходимое и перспективное направление развития предложенного алгебраического аппарата.

СПИСОК ЛИТЕРАТУРЫ

1. Данные в языках программирования: абстракция и типология. Сб. статей / Под ред. В. Агафонова. — М.: Мир, 1982. — 328 с.
2. Турский В. Методология программирования. — М.: Мир, 1981. — 264 с.
3. Шнейдерман Б. Психология программирования: человеческие факторы в вычислительных и информационных системах. — М.: Радио и связь, 1984. — 304 с.
4. Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра. Языки. Программирование. — Киев: Наук. думка, 1978. — 319 с.
5. Дорошенко А.Е., Акуловский В.Г. Алгебра алгоритмов с данными и прогнозирование вычислительного процесса // Проблеми програмування. — 2011. — № 3. — С. 3–10.
6. Акуловский В.Г. Основы алгебры алгоритмов, базирующейся на данных // Там же. — 2010. — № 2–3. — С. 89–96.
7. Акуловский В.Г. Алгебра для описания данных в композиционных схемах алгоритмов // Там же. — 2012. — № 2–3. — С. 234–240.
8. Дорошенко А.Е., Акуловский В.Г. Нисходящее проектирование алгоритмов в рамках алгеброалгоритмического подхода // Математические машины и системы. — 2012. — № 3. — С. 97–102.
9. Дорошенко А.Ю., Акуловський В.Г. Висхідне проектування алгоритмів при алгеброалгоритмичному підході // Вісн. Київ. нац. ун-ту імені Тараса Шевченка. Сер. Фіз.-мат. науки. — 2012. — Вип. 1. — С. 167–172.
10. Акуловский В.Г. Некоторые аспекты формализации архитектурного этапа разработки алгоритмов // Проблеми програмування. — 2009. — № 2. — С. 3–11.
11. Акуловский В.Г. Основы алгебры алгоритмов, базирующейся на данных // Мат. сьомої міжнар. наук.-практ. конф. з програмування УкрПРОГ'2010. Проблеми програмування. — 2010. — № 2–3. — С. 89–96.

Поступила 07.02.2013