# Некоторые аспекты формализации потоков данных в композиционных схемах алгоритмов

В. Г. Акуловский

*Академия таможенной службы Украины, ул. Дзержинского 2\4, Днепропетровск, Украина*

valeryakulovskiy@rambler.ru

**Аннотация.** *Одной из форм описания алгоритмов является композиционные схемы, введенные в рамках предложенной автором системы алгоритмических алгебр с данными. Композиционные схемы представляют собой композиции Д-операторов, на входе и выходе которых специфицированы обрабатываемые данные. Такая форма записи алгоритмов позволила формализовать информационные связи между Д-операторами. С целью распараллеливания алгоритмов введено понятие потоков данных в композиционных схемах. На основе выполненной формализации предполагается осуществлять преобразования композиционных схем для рационального распараллеливания алгоритмов на всех этапах их разработки.*

## Ключевые слова

Алгебра алгоритмов с данными, композиционные схемы алгоритмов, потоки данных в алгоритмах, распараллеливание алгоритмов

## 1 Введение

В литературе, в которой говорится о параллельных вычислениях, в первую очередь и главным образом речь идет о вычислительных алгоритмах. Такое положение вещей естественно, так как именно вычислительные алгоритмы сформировали высокие требования к вычислительной мощности и именно для реализации таких алгоритмов были разработаны суперкомпьютеры.

В настоящее время, когда многопроцессорными стали персональные компьютеры, спрос на параллельные алгоритмы возрастает принципиально. В этих условиях системы, ориентированные на разработку прикладного программного обеспечения, должны поддерживать разработку параллельных алгоритмов для широкого класса программных систем. В частности, в результате преобразование последовательной их реализации к параллельному виду. В качестве методологической основы систем разработки алгоритмов и программ может быть использован и используется алгеброалгоритмический подход [1].

## 2 Композиционные схемы алгоритмов

В рамках упомянутого подхода в результате модификации известной [2] модели ЭВМ Глушкова, модификация которого состояла в оснащении указанной модели ЭВМ памятью – носителем данных, автором в [3,4] была предложена алгебра алгоритмов с данными. Упомянутый формальный аппарат это - система алгоритмических алгебр (САА/Д) – $< \mathrm{U}, \mathrm{L}, \Omega >$, где $\mathrm{U}$ – множество Д-операторов; $\mathrm{L}$ – множество логических условий, $\Omega$ – сигнатура, состоящая из логических операций – $\Omega_1$, принимающих значения на множестве L, и операций – $\Omega_2$, принимающих значения на множестве $\mathrm{U}$.

Одной из форм описания последовательных алгоритмов на некоторых этапах их разработки является композиционные схемы (КС), представляющие собой описание алгоритма в виде композиции Д-операторов

$$(\mathrm{D})\mathrm{O}(\mathrm{D}') = (\mathrm{D}_1)\mathrm{O}_1(\mathrm{D}'_1) * (\mathrm{D}_2)\mathrm{O}_2(\mathrm{D}'_2) * \ldots * (\mathrm{D}_i)\mathrm{O}_i(\mathrm{D}'_i) * \ldots * (\mathrm{D}_n)\mathrm{O}_n(\mathrm{D}'_n),$$

где Д-оператор $(\mathrm{D}_i)\mathrm{O}_i(\mathrm{D}'_i)$, обрабатывает множество данных $\mathrm{D}_i$, специфицированных на его входе, и продуцирует множество данных $\mathrm{D}'_i$, специфицированных на его выходе, в качестве результата обработки. Обработка данных $\mathrm{D}'_i$, полученных в результате обработки $\mathrm{D}_i$ Д-оператором $(\mathrm{D}_i)\mathrm{O}_i(\mathrm{D}'_i)$, может быть и обычно продолжается некоторым другим Д-оператором $(\mathrm{D}_j)\mathrm{O}_j(\mathrm{D}'_j)$ и т.д. То есть исходные данные $\mathrm{D}_i$ до получения некоторого, в частности промежуточного, результата проходят (могут проходить) несколько этапов обработки. Д-оператор $(\mathrm{D}_j)\mathrm{O}_j(\mathrm{D}'_j)$, продолжающий обработку данных $\mathrm{D}_i$ на некотором этапе, информационно связан со всеми Д-операторами, принимавшими участие в обработке этих данных. Спецификация данных на входе и выходе Д-операторов явилась предпосылкой для формализации информационных связей, а КС – удобной формой для их описания.

# 3 Информационные связи в композиционных схемах алгоритмов

Рассмотрение начнем с определения понятия информационной связи.

**Определение 1.** Д-операторы $(D_i)O_i(D'_i)$ и $(D_j)O_j(D'_j)$ (i<j), входящие в КС $(D)O(D') = (D_1)O_1(D'_1) * \ldots * (D_i)O_i(D'_i) * \ldots * (D_j)O_j(D'_j) * \ldots * (D_n)O_n(D'_n)$, связаны, если для них выполняется соотношение $D'_i \cap D_j \neq \varnothing$, а данные, образующие множество $D'_i \cap D_j$, называются связывающими. При этом данные ${}_i\overleftarrow{D}_j$ на входе j-го Д-оператора, связывающие его с i-ым такие, что $D_j \supseteq {}_i\overleftarrow{D}_j = D'_i \cap D_j$, и данные ${}_j\overrightarrow{D}_k$ на выходе j-го Д-оператора, связывающие его с k-ым такие, что $D'_j \supseteq {}_j\overrightarrow{D}_k = D'_j \cap D_k$, называются, соответственно, левой и правой связью Д-оператора $(D_j)O_j(D'_j)$. Связи вида ${}_i\overleftarrow{D}_{i+1}$ и ${}_i\overrightarrow{D}_{i-1}$ будем называть прямыми.

Поскольку все предшествующие Д-операторы могут быть связаны с данным, а он со всеми последующими, введем такое определение.

**Определение 2.** Множество $S^{л}_j = {}_1\overleftarrow{D}_j, {}_2\overleftarrow{D}_j, .., {}_l\overleftarrow{D}_j, , \ldots, {}_{j-1}\overleftarrow{D}_j$, назовем множеством левых связей Д-оператора $(D_j)O_j(D'_j)$, а множество $S^{пр}_j = {}_j\overrightarrow{D}_{j+1}, {}_j\overrightarrow{D}_{j+2}, \ldots, {}_j\overrightarrow{D}_p, \ldots, {}_j\overrightarrow{D}_n$ - множеством его правых связей.

Введенное определение позволяет специфицировать все связи, имеющие место в КС. Для самого общего случая, когда все Д-операторы, входящие в КС, связаны между собой, то есть каждый Д-оператор связан со всеми следующими за ним и всеми предшествующими ему Д-операторами, КС выглядит следующим образом.

$$(D)O(D') = (D_1)O_1(D'_1, {}_1\overrightarrow{D}_2, {}_1\overrightarrow{D}_3, \ldots, {}_1\overrightarrow{D}_j, \ldots, {}_1\overrightarrow{D}_n) * (D_2, {}_1\overleftarrow{D}_2,)O_2(D'_2, {}_2\overrightarrow{D}_3, \ldots, {}_2\overrightarrow{D}_j, \ldots, {}_2\overrightarrow{D}_n) * \ldots$$

$$\ldots * (D_j, {}_1\overleftarrow{D}_j, .., {}_l\overleftarrow{D}_j, , \ldots, {}_{j-1}\overleftarrow{D}_j)O_j(D'_j, {}_j\overrightarrow{D}_{j+1}, \ldots, {}_j\overrightarrow{D}_p, \ldots, {}_j\overrightarrow{D}_n) * \ldots * (D_n, {}_1\overleftarrow{D}_n, .., {}_j\overleftarrow{D}_n, \ldots, {}_{n-1}\overleftarrow{D}_n)O_n(D'_n).$$

В данном случае не только специфицированы связи между операторами, но и все источники и приемники данных поставлены в однозначное соответствие, то есть любому ${}_m\overrightarrow{D}_p$ соответствует ${}_m\overleftarrow{D}_p$. Поскольку в КС, как правило, не все операторы связаны друг с другом, любое из множеств ${}_m\overrightarrow{D}_p$ может быть пустым и, соответственно, пустым будет и множество ${}_m\overleftarrow{D}_p$.

Исходя из определения, будем различать три типа Д-операторов, которые назовем:

- связанные, если $S^{л}_j \neq \varnothing$ и $S^{пр}_j \neq \varnothing$;
- связанные слева (справа), если $S^{л}_j \neq \varnothing$, а $S^{пр}_j = \varnothing$ ($S^{пр}_j \neq \varnothing$, а $S^{л}_j = \varnothing$);
- несвязанные, у которых $S^{л}_j = \varnothing$ и $S^{пр}_j = \varnothing$.

Формализация информационных связей представляет самостоятельный интерес с точки зрения анализа и преобразования алгоритмов. В [5] показана, например, возможность сокращения длины информационных связей в КС. Не менее важным, однако, является то обстоятельство, что Д-операторы $O_i, O_j, O_{j+1}$, находящиеся в отношении связанные $O_i \prec O_j$, в частности прямо связанные $O_j < O_{j+1}$, упорядочены в множестве Д-операторов

$$\{(D_1)O_1(D'_1), \ldots, (D_i)O_i(D'_i), \ldots, (D_j)O_j(D'_j), (D_j)O_j(D'_j), \ldots, (D_n)O_n(D'_n)\},$$

образующих КС. Поскольку связи между Д-операторами определяют порядок их выполнения, для решения задачи распараллеливания алгоритмов представляют интерес совокупности (последовательности) информационных связей в КС, которые образуют информационные потоки.

# 4 Информационные потоки в композиционных схемах алгоритмов

В дальнейших рассуждениях будем рассматривать только правые связи, множество которых обозначим $D^{св}$.

Будем говорить, что связывающие данные ${}_{i}\vec{D}_{j} \subset D^{св}$ поступают с выхода Д-оператора $O_i$, который является "истоком" связывающих данных с адресом i, на вход Д-оператора $O_j$, который является "стоком" связывающих данных с адресом j.

В связи с этим будем утверждать следующее.

**Утверждение.** Максимальное количество правых связей в композиционной схеме

$$(D)O(D') = (D_1)O_1(D'_1,{}_1\vec{D}_2,{}_1\vec{D}_3,...,{}_1\vec{D}_j,...,{}_1\vec{D}_n)*(D_2)O_2(D'_2,{}_2\vec{D}_3,...,{}_2\vec{D}_j,...,{}_2\vec{D}_n)*...$$

$$...*(D_j)O_j(D'_j,{}_j\vec{D}_{j+1},...,{}_j\vec{D}_p,...,{}_j\vec{D}_n)*...*(D_n)O_n(D'_n),$$

то есть мощность множества всех правых связей $|D^{св}|$, равна $(n^2-n)/2$.

Доказательство. Так как для первого истока количество правых связей равно $n-1$, для второго $n-2$ и т.д, для предпоследнего истока $n-(n-1)$, то легко увидеть, что $|D^{св}|=(n^2-n)/2$.

Теперь определим понятие поток данных.

**Определение 3.** Упорядоченная выборка ${}_{j_1}\vec{D}_{j_2},{}_{j_2}\vec{D}_{j_3},...,{}_{j_{k-1}}\vec{D}_{j_k}$ из множества связывающих данных $D^{св}=\{{}_1\vec{D}_2,{}_1\vec{D}_3,...,{}_1\vec{D}_j,...,{}_1\vec{D}_n,{}_2\vec{D}_3,...,{}_2\vec{D}_j,...,{}_2\vec{D}_n,...,{}_j\vec{D}_{j+1},...,{}_j\vec{D}_p,...,{}_j\vec{D}_n,...,{}_{n-1}\vec{D}_n\}$ такая, что связанные этими данными Д-операторы образуют упорядоченную выборку $O_{j_1},O_{j_2},...,O_{j_k}$ из множества $O=O_1,O_2,...,O_j,...,O_n$, называется потоком данных и обозначается $\vec{D}_i^{п}$. Поток, содержащий единственный элемент ${}_{j_i}\vec{D}_{j_{i+1}}$, назовем элементарным. Д-операторы, "попавшие" в поток данных, будем называть узлами потока.

Поскольку в КС имеют место прямые связи, введем следующее определение.

**Определение 4.** Поток, содержащий прямые и только прямые связи, то есть поток вида ${}_j\vec{D}_{j+1},{}_{j+1}\vec{D}_{j+2},...,{}_{j+k}\vec{D}_{j+k+1}$, назовем прямым потоком данных.

Рассмотрев множество потоков данных $\vec{D}^{П}=\vec{D}_1^{П},\vec{D}_2^{П},...\vec{D}_p^{П}$ для случая, когда в качестве первого потока выбран прямой поток данных вида $\vec{D}_1^{П}={}_1\vec{D}_2,{}_2\vec{D}_3,...,{}_j\vec{D}_{j+1},...,{}_{n-1}\vec{D}_n$, увидим, что при выборе каждого следующего потока максимальной длины, в состав $\vec{D}^{П}$ входят следующие потоки данных:

$\vec{D}_2^{П}={}_1\vec{D}_3,{}_3\vec{D}_5,...,{}_{n-2}\vec{D}_n$;

$\vec{D}_3^{П}={}_1\vec{D}_4,{}_4\vec{D}_7,...,{}_{n-3}\vec{D}_n$;

…

$\vec{D}_{p-1}^{П}={}_{n-3}\vec{D}_{n-1},{}_{n-3}\vec{D}_n$;

$\vec{D}_p^{П}={}_{n-2}\vec{D}_n$.

Из этого простого опыта видно, по крайней мере, следующее:

- состав множества потоков не является единственным;
- выбор очередного потока влияет на состав потоков в КС;
- на состав потоков влияет длина выбираемых потоков данных, которая сокращается с каждым сделанным выбором;
- последний поток оказался элементарным;
- одни и те же узлы могут попадать как в один, так и в несколько потоков данных.

Кроме того, из изложенного известно, что:

- \- состав потоков определяется данными, специфицированными на входах и выходах Д-операторов;
- \- наличие и количество несвязанных или связанных слева (справа) Д-операторов влияет на конфигурацию потоков.

# 5 Заключение

Из изложенного можно сделать следующие очевидные выводы:

- КС с приведенным прямым потоком данных является граничным случаем и не может быть распараллелена.

- Вторым граничным случаем является КС, в которой все Д-операторы несвязанные. В этом случае все Д-операторы, входящие в КС, могут выполняться параллельно.
- Если узел попадает в два или более потоков, то эти потоки данных нуждаются в синхронизации.

В достаточно обширном диапазоне, лежащем между указанными граничными случаями, открываются возможности к распараллеливанию КС. Будем полагать, что изучение всех возможных потоков данных в КС позволит с учетом того, что информационные потоки упорядочивают множество Д-операторов, входящих в КС, выявить естественный параллелизм, присущий алгоритму. Анализ связей между узлами позволит реализовать синхронизацию параллельно выполняемых потоков и оптимизировать объем информации, необходимый для их взаимодействия. Весьма существенным является тот факт, что, при реализации стратегии нисходящего проектирования, распараллеливание алгоритма может выполняться на всех этапах его декомпозиции, на которых алгоритм записывается в виде КС.

## Список литературы

[1] Ф.И. Андон, А.Е. Дорошенко, Г.Е. Цейтлин, Е.А. Яценко. Алгеброалгоритмические модели и методы параллельного программирования. - К.:Академперіодика, 2007.-634 с.
[2] Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра. Языки. Программирование. -К.: Наукова думка, 1978.-319с.
[3] Акуловский В.Г. Основы алгебры алгоритмов, базирующейся на данных. Матеріали сьомої міжнародної науково-практичної конференції з програмування УкрПРОГ`2010. Київ.-2010. В журнале Проблеми програмування. - 2010.-№2-3.-С.89-96.
[4] Акуловский В.Г. Некоторые аспекты формализации данных и декомпозиция Д-операторов алгебры алгоритмов// Проблеми програмування.- 2009.-№4-С.3-10
[5] Акуловский В.Г. Некоторые аспекты преобразования алгоритмов на основе формализации информационных связей// Кибернетика и системный анализ.-2009.-№6.-С.50-54.