УДК 681.324

Представление операторов выбора и цикла языков программирования в граф-схемах алгоритмов

© Ю.М. Руденко

МГТУ им. Н.Э. Баумана, Москва, 105005, Россия

Рассмотрена возможность использования граф-схем для изображения алгоритмов решения параллельных задач при наличии циклических вычислений, а также условных операторов выбора направления вычислений. Предложенные методы изображения данных операторов позволяют легко строить параллельные ветви алгоритмов, что является достоинством этих методов. Отмечены различия для наиболее часто используемых языков программирования.

Ключевые слова: *вычислительная система, ВС, элементарный процессор, ЭП, оператор выбора, оператор цикла, граф-схема алгоритма, оператор цикла по параметру, языки программирования, параллельные ветви алгоритмов, ребро графа.*

В настоящее время все более широкое применение получило представление алгоритмов для параллельных вычислений в виде граф-схем. Одной из первых работ этого направления следует считать [1]. Дальнейшее развитие данный подход получил в работах [2, 3, 4, 5, 6 и др.]. Как известно, основная идея этого подхода — представление программных модулей (процедур, функций и других подобных конструкций) решаемой на вычислительной системе задачи в виде взвешенных вершин граф-схемы. Вес вершины может определять, например, время выполнения соответствующего программного модуля. Связи между программными модулями изображаются в виде дуг, возможно с весами, определяемыми объемами передаваемых данных. Существуют модификации этого направления, предполагающие использование сетей Петри [7] и использование элементов теории графов для создания параллельных алгоритмов при решении систем линейных алгебраических уравнений трехдиагонального вида методом встречных прогонок. В работе [8] выбран оптимальный алгоритм среди известных и представленных алгоритмов с локальными коммуникациями между задачами.

Так как построенная граф-схема [4] в дальнейшем используется для создания параллельных ветвей алгоритма, при написании программных модулей операторы условного перехода, выбора, цикла по счетчику циклов и некоторые операторы цикла по параметрам целесообразно показывать в граф-схеме алгоритма решения задачи. В настоящей работе для удобства изложения материала сделано разделение между операторами условного перехода и выбора, хотя в лите-

ратуре оба понятия часто обобщаются. С этой целью при формировании схемы алгоритма в соответствии с ГОСТ 19.701-90 ЕСПД необходимо для каждого выше перечисленного оператора создавать фрагмент граф-схемы с учетом особенности выполнения названных операторов в соответствующем языке программирования. В связи с тем, что языков программирования достаточно много, далее будут рассматриваться только наиболее часто используемые, такие как C [10], C++ [10], C# [11], Java [12], Pascal [13], Delphi [14], Visual Basic [15], VB.NET [15], Fortran 90 [16].

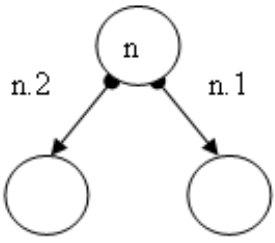

Рис. 1. Представление условного оператора в граф-схеме для языков программирования C, C++, C#, Java, Pascal, Delphi, Visual Basic, VB.NET, Fortran 90

К настоящему моменту оператор условного перехода унифицирован для всех перечисленных языков, т. е. имеет два выхода — «истина» и «ложь». Поэтому в граф-схеме алгоритма он реализуется в виде фрагмента граф-схемы, показанного на рис. 1, где n — это номер вершины в граф-схеме, n.1, n.2 определяют номера выходов. Такое обозначение принято в связи с тем, что эти номера могут использоваться в матрицах анализа граф-схем, например, в матрице следования [9]. При необходимости для понимания смысла граф-схемы обычно составляют таблицу номеров выходов с пояснением выполняемых ими функций.

Операторы выбора для языков C++, C#, Java, Pascal, Delphi, Visual Basic, VB.NET, Fortran 90, несмотря на их различные синтаксисы изображаю одинаково, как показано на рис. 2 .

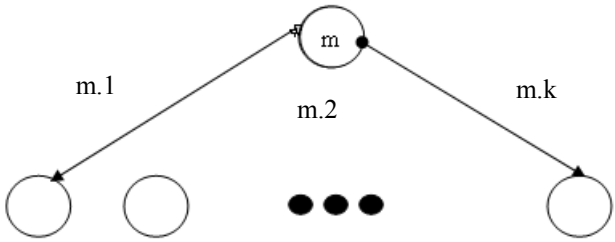

Рис. 2. Представление операторов выбора в граф-схеме языков программирования C, C++, C#, Java, Pascal, Delphi, Visual Basic, VB.NET, Fortran 90

На языке C, используя оператор «break», изменяется фрагмент граф-схемы, изображающей оператор «switch». При отсутствии оператора «break» в некотором операторе «case» выполняются все очередные операторы «case» до встречи очередного оператора «break»

или до конца оператора. На рис. 3 показан фрагмент граф-схемы, изображающий оператор «switch» (при отсутствии в нем операторов «break» в операторе «case»), который соответствует дуге с номером m.k–1.

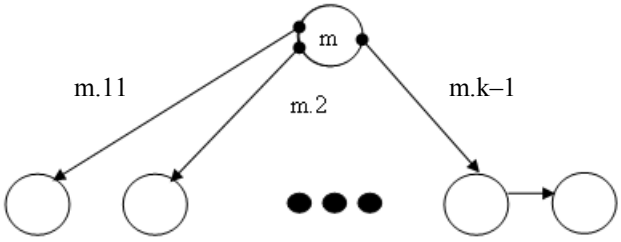

Рис. 3. Фрагмент граф-схемы оператора «switch», в котором отсутствует оператор «break» в операторе «case» с номером m.k–1

Для изображения циклов в граф-схемах алгоритмов рассматриваются три типа циклов. *Первый тип* — циклы по счетчику циклов, в которых известно количество итераций. Изображение фрагмента граф-схемы для этого случая представлено на рис. 4, где время выполнения одной итерации t [17]. Если количество повторений цикла больше числа отводимых для вычислений цикла ЭП (в данном случае — k), тогда каждый i-й ЭП выполняет несколько итераций с временем $T_i = ts$, где s — количество итераций, выполняемых каждым элементарным процессором, $T_i \in \{T_1, T_2, \ldots, T_k\}$. В случае, когда в результате выполнения цикла получается одно значение, фрагмент граф-схемы оператора цикла по счетчику циклов выглядит, как показано на рис. 5.

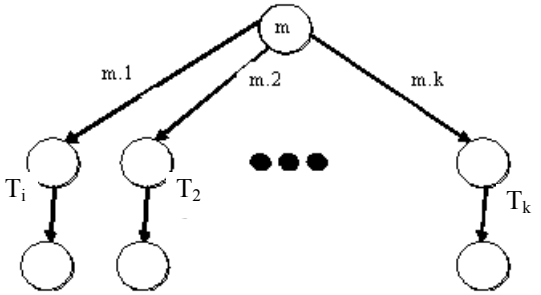

Рис. 4. Фрагмент граф-схемы оператора цикла по счетчику циклов, в котором количество итераций в цикле больше числа использованных для организации цикла элементарных процессов

Второй тип циклов связан с параметром, значение которого определяет момент окончания цикла. Особенностью данного типа является возможность преобразования его к виду, удобному для вы-

числения его значения одновременно на нескольких элементарных процессорах. Фрагмент граф-схемы для этого случая аналогичен фрагменту, представленному на рис. 5.

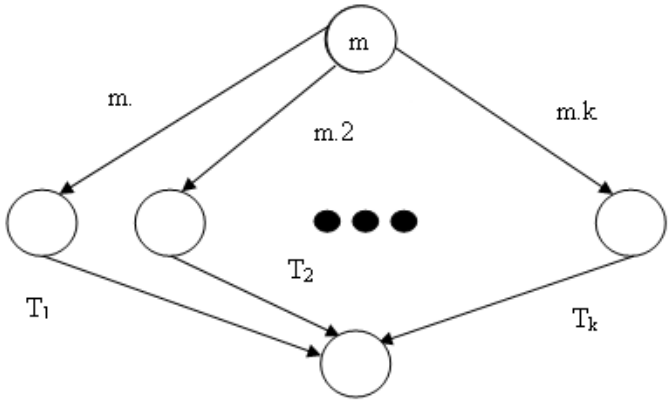

Рис. 5. Фрагмент граф-схемы оператора цикла по счетчику циклов, в котором количество итераций в цикле больше числа использованных для организации цикла элементарных процессов (в результате выполнения цикла вычисляется одно значение)

В качестве примера можно привести вычисление определенного интеграла на интервале [a, b] с заданной точностью ε. Интервал [a, b] делим на n составляющих рассматриваемого интервала, где n — некоторое целое число. Вычисление интеграла в этом случае можно выполнить одновременно на n ЭП и затем произвести суммирование полученных значений.

Третий тип циклов связан также с параметром, но метод решения этого типа нельзя свести к предыдущему случаю. Цикл в этом случае включается в программный модуль и вычисляется на одном элементарном процессоре.

Приведем пример использования граф-схемы для представления алгоритма решения некоторой задачи. Здесь следует сделать замечание о том, что содержательная часть задачи нас интересует только с точки зрения наличия в ней условных операторов, операторов выбора и циклов. Также будем считать, что все необходимые данные для вычислений находятся в соответствующей локальной памяти элементарного процессора. Пример схемы алгоритма в соответствии с ГОСТ 19.701-90 ЕСПД, которая затем будет преобразована в граф-схему, представлен на рис. 6. На этой схеме показан оператор выбора *2*, подпрограммы *3–5*, которые вычисляют некоторые функции F_1, F_2, F_3. Подпрограмма *6* будет реализована, например, на пяти элементарных процессорах. Предполагается, что функция f (x) удовлетворяет условиям численного интегрирования.

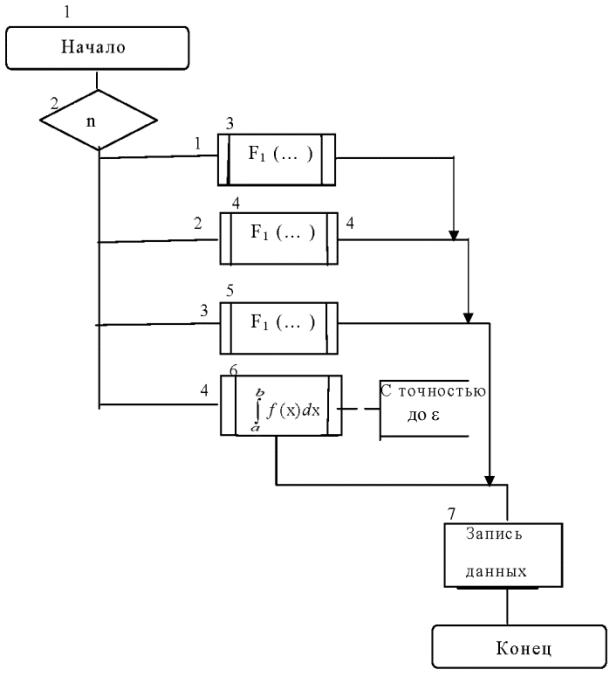

Рис. 6. Схема алгоритма, представленная в соответствии с ГОСТ 19.701-90 ЕСПД, которая будет преобразована в граф-схему (с ее помощью организуются параллельные вычисления)

Реализация алгоритма решения задачи в виде граф-схемы показана на рис. 7, на котором вершины перенумерованы в соответствии с ярусами графа. Такая нумерация не редкость и позволяет использо-

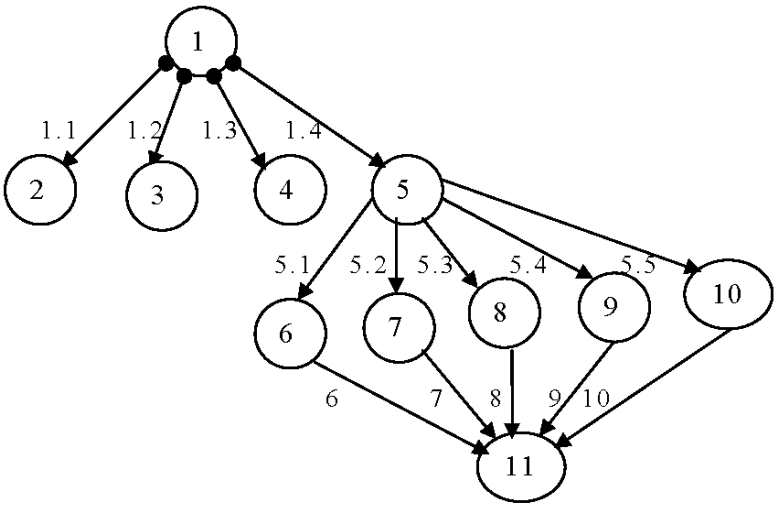

Рис. 7. Граф-схема алгоритма, представленного на рис. 6 в виде схемы алгоритма в соответствии с 19.701-90 ЕСПД

вать для анализа граф-схем треугольные матрицы. В достаточно простых граф-схемах около вершин проставляются их веса. Вес вершин определяет, например, время выполнения программного модуля, который идентифицируется рассматриваемой вершиной. На связях граф-схем помимо номера связи или вместо нее указывается время передачи информации. В более сложных случаях целесообразно проставлять номера вершин и связей и, используя эти номера, составлять таблицу комментариев, в которой представляются все необходимые сведения.

№ вершины 1	Время выполнения T_1	Функциональное назначение. Проверка условия оператора выбора
№ связи 1.1	Время выполнения t_1	Функциональное назначение. Переход при выполнении первого условия
…		
№ связи 1.4	Время выполнения t_4	Функциональное назначение. Переход при выполнении четвертого условия
№ вершины 5	Время выполнения T_2	Функциональное назначение. Разбиение интервала [a, b] на пять частей
№ связи 1.5	Время выполнения t_5	Функциональное назначение Передача 1-го интервала на 6-й элементарный процессор
…		
№ вершины 11	Время выполнения T_{11}	Функциональное назначение. Вычисление численного значения интеграла

Таким образом, в данной работе предлагается развитие методов применения теории графов для изображения параллельных алгоритмов решения сложных задач с помощью вычислительных систем.

ЛИТЕРАТУРА

[1] Барский А.Б. *Параллельные процессы в вычислительных системах*. Москва, Радио и связь, 1990, 256 с.

[2] Водяхо А.И., Горнец Н.Н., Пузанков Д.В. *Высокопроизводительные системы обработки данных*. Москва, Высш. шк., 1997, 150 с.

[3] Воеводин В.В., Воеводин Вл.В. *Параллельные вычисления*. Санкт-Петербург, БХВ-Петербург, 2002, 600 с.

[4] Руденко Ю.М. Представление параллельных алгоритмов в виде граф-схем. Аэрокосмические технологии. *Научные материалы Международной научно-технической конференции*—2009. Реутов—Москва 2009. Москва, 177–179 с.

[5] Гергель В.П. *Теория и практика параллельных вычислений*. Москва, Бином, Лаборатория знаний, 2007. 424 с.
[6] Цилькe Б.Я., Орлов С.А. *Организация ЭВМ и систем*. Санкт-Петербург, Питер, 2007, 672 с.
[7] Васильев В.В., Кузьмук В.В. *Сети Петри, параллельные алгоритмы и модели мультипроцессорных систем*. Киев, Наукова думка, 1990, 216 с.
[8] Головашкин Д.Л. Применение метода встречных прогонок для синтеза параллельного алгоритма решения сеточных уравнений трехдиагонального вида. Самара, Институт систем обработки изображений РАН, Самарский государственный аэрокосмический университет.
URL: КО_PDF_12024_КО24105.PDF
[9] Руденко Ю.М. Учет зависимостей программных модулей по данным и последовательностям их выполнения при параллельных вычислениях. *Известия высших учебных заведений*. Поволжский регион. Технические науки, 2009, № 3 (11), с. 67–75.
[10] Березин Б.И., Березин С.Б. *Начальный курс С и С++*. Москва, ДИАЛОГ-МИФИ, 2001, 288 с.
[11] Робинсон У. *C# без лишних слов*. Москва, ДМК Пресс, 2002, 340 с.
[12] Арнольд К., Гослинг Д. *Язык программирования JAVA*. Москва, Питер, 1997, 250 с.
[13] Гордон Я. *Тонкости программирования на языке Паскаль:* учебное пособие по программированию на ПК. Москва, Бук-Пресс, 2006, 320 с.
[14] Архангельский А.Я. Программирование в Delphi для Windows. Версии 2006, 2007. Москва, Бином-Пресс, 2007, 1248 с.
[15] Атли К. А *Programmer's Introduction to Visual Basic*. NET. Москва, ДМК Пресс, 2004, 3004 с.
[16] Меткалф М., Рид Дж. *Описание языка программирования Фортран 90*. Москва, Мир, 1995, 302 с.
[17] Мусина Л.В., Руденко Ю.М. Временная задержка на вычислительных модулях при реализации граф-схем. *Вестник МГТУ им. Н.Э. Баумана. Сер. «Приборостроение»*. Спецвыпуск «Информационные технологии и компьютерные системы», 2011, с. 71–75.

Статья поступила в редакцию 28.06.2013

Ссылку на эту статью просим оформлять следующим образом:
Руденко Ю.М. Представление операторов выбора и цикла языков программирования в граф-схемах алгоритмов. *Инженерный журнал: наука и инновации*, 2013, вып. 11. URL: http://engjournal.ru/catalog/it/hidden/1066.html

Руденко Юрий Михайлович родился в 1941 г., окончил Харьковский политехнический институт в 1964 г. Канд. техн. наук, доцент кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана. Автор около 100 опубликованных работ в области компьютерных технологий. e-mail: kirur@bk.ru