05.13.11

ВИЗУАЛЬНО-ГРАФИЧЕСКАЯ СИСТЕМА ПРОГРАММИРОВАНИЯ НА ОСНОВЕ РАЗРАБОТКИ БЛОК-СХЕМ АЛГОРИТМОВ. ПРОСТРАНСТВЕННОЕ РАЗМЕЩЕНИЕ СИМВОЛОВ-БЛОКОВ СЛОЖНЫХ АЛГОРИТМОВ

О.А. Евдокимова, А.А. Вохмин, А.А. Малявко
НГТУ, г. Новосибирск

В работе приведены результаты исследования и разработки редактора блок-схем алгоритмов. Редактор представляет собой один из компонентов проектируемой многоязыковой визуально-графической системы программирования. При ее разработке приходится решать ряд сложных задач, в числе которых – задача автоматического размещения текущего представления элементов (блоков, контейнеров) и связей между ними, в окне, наложенном на потенциально значительно большую по размерам блок-схему проектируемого алгоритма. Способ представления (свернутый или развернутый) каждого контейнера блок-схемы, возможно, содержащего десятки или сотни вложенных в него элементов, на каждом шаге проектирования алгоритма, может выбираться пользователем.

Предложен и описан способ пространственного размещения элементов/контейнеров сложных алгоритмов в автоматическом режиме с учетом необходимости реализации всех потенциально возможных операций пользователя по модификации проектируемой блок-схемы. Рассмотрены необходимые для реализации этого подхода внутренние структуры данных визуально-графического редактора и некоторые наиболее важные алгоритмы расчета/пересчета координат блоков/контейнеров и связей между ними в процессе автоматического размещения.

Ключевые слова: визуальный редактор, блок-схема, алгоритм, пространственное размещение, расчет координат.

ВВЕДЕНИЕ

Разработка действительно сложных алгоритмов очень слабо автоматизирована. Обычным в настоящее время является подход, предлагаемый объектно-ориентированной парадигмой и состоящий в разбиении единого видения сложного процесса обработки информации на большое количество взаимодействующих друг с другом классов, каждый из которых содержит множество интерфейсов, методов и полей. С одной стороны, такой подход позволяет существенно уменьшить сложность каждого отдельного объекта, принимающего участие в решении единой задачи и тем самым упростить процессы его проектирования, реализации в программном коде, отладки и сопровождения. С другой стороны, за деревьями становится не видно леса. Множество деталей, включаемых в программный проект в соответствии с требованиями объектно-ориентированной технологии, может значительно заслонить существо единого алгоритма и затруднить процесс его разработки.

Поэтому представляет интерес исследование и разработка альтернативных подходов. Одним из них является представление и разработка сложного алгоритма в виде одной или совокупности блок-схем с использованием соответствующих потребностям разработчика средств автоматизации визуального представления программного проекта. Для функционирования таких средств необходимы мощные вычислители и графические процессоры, их отсутствие сдерживало развитие этого подхода. В настоящее время существующие технические средства позволяют обеспечивать необходимую производительность подготовки и обработки графической информации.

Проектирование алгоритма на основе блок-схем можно представить себе как процесс постепенного преобразования начального представления в виде единственного блока во все более и более детализированное графическое представление, содержащее функциональные и управляющие элементы, объединенные связями по управлению. Каждый шаг такого преобразования может состоять в замене одного функционального блока на последовательность из двух, трех, и т.д. блоков (например, вместо «инициализация массивов» появляется «заполнение матрицы заданными значениями» и «очистка вектора»), либо в замене функционального блока на управляющий (например, вместо блока «вычисление суммы двух векторов» появляется совокупность блоков, реализующих «цикл вычисления сумм элементов двух векторов»).

При этом важно сохранять возможность отображения начального представления блок-схемы (но полного сохранения всех деталей, внесенных в процессе преобразования). Важно также снять с разработчика алгоритма заботу о пространственном размещении элементов разрабатываемой блок-схемы. Это актуально потому, что решать эту непростую

задачу необходимо буквально на каждом шаге детализации проектируемой блок-схемы.

Такой подход предполагает разработку визуально-графического редактора блок-схем алгоритмов. Далее рассматриваются задачи, которые должен решать этот редактор и некоторые способы их решения.

ПОСТАНОВКА ЗАДАЧИ

Цель работы – разработать способ пространственного размещения символов-блоков сложных алгоритмов в автоматическом режиме и реализовать данный способ в редакторе.

Для достижения данной цели необходимо выполнить следующие задачи:

1. Определить требования к визуальному представлению сложных алгоритмов в виде блок-схем.
2. Провести исследование с целью изучения существующих методов визуализации блок-схем.
3. Разработать способ пространственного отображения символов блоков, удовлетворяющий заявленным критериям.
4. Определить инструменты программной реализации разработанного алгоритма.
5. Реализовать алгоритм пространственного размещения символов-блоков.

ТРЕБОВАНИЯ К ВИЗУАЛЬНОМУ ПРЕДСТАВЛЕНИЮ СЛОЖНЫХ АЛГОРИТМОВ В ВИДЕ БЛОК-СХЕМ

Разрабатываемый алгоритм будет осуществлять работу со сложными, массивными программами. Это значит, что необходимо учитывать размер программы при отображении схемы на экране (баланс возможности прочесть представленную информацию и уместить визуальное представление на экране).

Описанное требование возможно выполнить механизмом сокрытия деталей реализации некоторых частей программы. Для просмотра деталей реализации пользователю предоставляется механизм управления процессом сокрытия некоторых деталей.

Так же важно, чтобы пользователь имел возможность выбирать, с какой степенью детальности он хочет просматривать алгоритм.

Важен механизм изменения масштаба представляемой схемы и просмотр некоторых ее частей отдельно (например, в другой вкладке или окне).

СУЩЕСТВУЮЩИЕ МЕТОДЫ ВИЗУАЛИЗАЦИИ БЛОК-СХЕМ

Первым шагом исследования существующих методов визуализации блок-схем был анализ имеющихся визуальных редакторов.

Таких редакторов реализовано большое количество. Многие из них требуют от пользователя размещения символов-блоков вручную, некоторые позволяют делать это автоматически. Для автоматического пространственного расположения блоков в таких редакторах нет механизма упрощения чтения схемы, то есть алгоритм отображается со всеми деталями на одном изображении. Возможность исследовать программную реализацию алгоритмов отрисовки редакторов-аналогов ограниченна, так как не многие имеют открытый исходный код.

Вторым шагом анализа существующих алгоритмов визуализации блок-схем был поиск исследований на данную тему. Особо интересным оказался подход, описанный в статье «Dynamically Generated Block Diagrams as a Visualisation Method»[1]. Данный способ визуализации был разработан для блок-схем, где есть возможность перетаскивать символы. В этой возможности нет необходимости в разрабатываемом способе, поэтому необходимо новое видение.

АЛГОРИТМ ПРОСТРАНСТВЕННОГО ОТОБРАЖЕНИЯ СИМВОЛОВ-БЛОКОВ

Разработанный алгоритм пространственного отображения символов-блоков состоит из следующих шагов:

1)связывание блоков;
2)сокрытие деталей реализации;
3)расчет координат блоков;
4)расчет координат контейнеров;

Связывание блоков протекает по следующему механизму. Первый блок схемы –вход в программу - узел, являющийся родительским для всех последующих блоков в схеме. Далее блоки связываются по принципу дерева – у узла есть один родитель и один потомок. У узла так же может быть сосед - в зависимости от типа блока (например, у блоков, реализующих логику операции условия, может быть сосед).

На связывание блоков влияет информация об уровне вложенности блока – значение, определяемое механизмом сокрытия деталей реализации.

На рисунке 1 представлена возможные логические связи блоков.

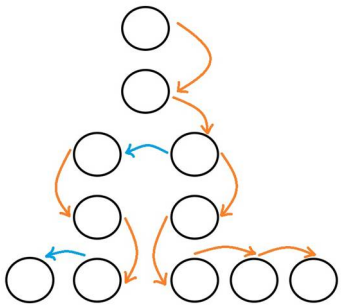

Рис. 1. Следование ссылок в алгоритме

Оранжевыми стрелками представлены связи блока с потомками, синими стрелками представлены соседи, находящиеся на одном уровне и имеющие одного родителя.

Блоки должны скрывать детали реализации (сворачиваться) до тех пор, пока на панели редактирования не останется один элемент. В зависимости от типа символа-блока, он может скрывать операции, подробно описывающие именно этот оператор.

Приведенную логику можно представить, как дерево – структуру данных, используемую для отображения иерархии и моделирования процессов принятия решений. [2]

Узлом в таком дерево будет являться фрагмент схемы, состоящий из совокупностей блоков, логика работы которых связана (описывают один процесс). Ветви – связи между блоками. Родитель для сворачиваемой совокупности блоков – совокупность блоков, предшествовавших текущему узлу. Потомки – последующая совокупность блоков. Соседи – блоки в скрываемом множестве.

Такое дерево будет являться упорядоченным (расположение дочерних элементов имеет значение) и несбалансированным (разница в высоте между поддеревьями может превышать единицу). Дерево не является двоичным по причине реализации оператора switch (имеет неограниченное количество ветвей потомков), но в большинстве случаев потомок будет один, реже – два (реализация оператора условия).

Одним из индексов определения последовательности в дереве будет уровень вложенности, измеряемый числом. Значения дочерних узлов вложенности будет всегда больше, чем у родительского узла (min-heap). Если бы дерево было бинарным и сбалансированным, его можно было бы назвать пирамидой, двоичной кучей или частично упорядоченным деревом. [3]

Графическое представления дерева вложенности представлено на рисунке 2. Приведенное изображение отображает логику задания родительских узлов и уровня вложенности.

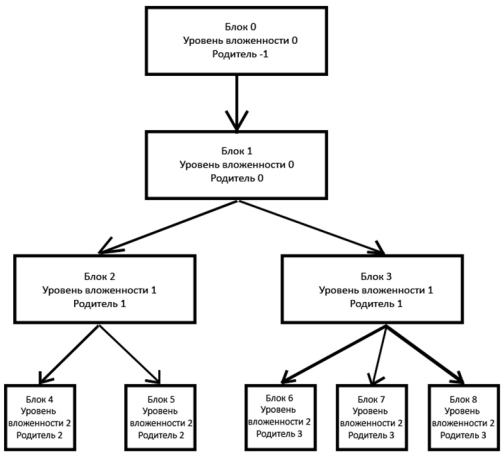

Рис. 2. Графическое представления дерева уровней вложенности

Поиск контейнеров, которые необходимо скрыть осуществляется обходом дерева в ширину.

Одним из вариантов решения задачи рассматривался формат реализации R-дерева [4]. Данный формат дерева базируется на разбиении данных исходя из имеющихся у них координат. Чаще всего алгоритм применяется для классификации объектов при работе с картами. Работа с координатами – подходящий случай, так как блоки размещаются в пространстве, но R-дерево является сбалансированным, автоматически разделяя блоки при превышении максимально допустимого количества узлов в одной из ветвей. Балансировка дерева может разрушить логическую последовательность выполнения программы, отображаемой в виде блок-схемы, поэтому данный алгоритм не подошел.

Изначально при вычислении координат символов-блоков не производится проверки на пересечение с другими символами. На шаге вычисления координат блоков, примитивам символов присваиваются координаты относительно родительского блока. Это временные координаты, которые изменяются при наличии пересечений в шаге расчета координат контейнеров.

Вычисление координат блоков основывается на пространственном расположении блока-родителя и размерах добавляемого блока. Размер можно вычислить исходя из типа блока (размер у каждого типа блока свой) – данное вычисление используется при добавлении нового блока. Или же размер можно получить из данных о блоке – используется, если координаты высчитываются для уже существующего блока.

Далее важно является ли блок первым на диаграмме. Если блок первый – он отрисовывается по центру панели редактирования. Иначе мы получаем координаты блока-родителя.

Следующий шаг – определение типа родительского блока. Если родительский блок подразумевает одного потомка, вычисление координат происходит как показано на рисунке 3.

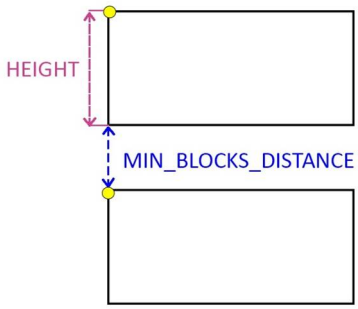

Рис. 3. Вычисление координат одного потомка

Блок имеет ту же координату по оси абсцисс, но по оси ординат смещается на константное значение MIN_BLOCKS_DISTANCE – минимальное расстояние между блоками.

Если родительский блок подразумевает двух потомков, производятся вычисления, отраженные на рисунке 4.

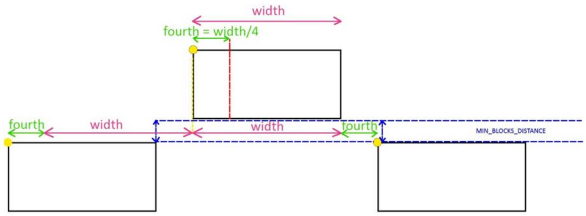

Рис. 4. Вычисление координат для двух потомков

Высчитывается четверть ширины родительского блока. Это расстояние и есть расстояние между блоками. Значение координат блоков задается для верхней левой точки блока, поэтому помимо смешения на четверть ширины родителя, необходимо так же сместиться и на ширину блока.

Данное вычисление – смещение по оси абсцисс, по оси ординат будет так же смещение на константу MIN_BLOCKS_DISTANCE.

Вычисление координат контейнера производится при каждом добавлении нового блока. Это позволяет не пересекаться отображениям блоков.

Сначала производится вычисление координат контейнера на основе блоков, вложенных в него:

– получить координаты самого верхнего левого контейнера;
– получить координаты самого правого контейнера;
– получить координаты контейнера, расположенного ниже всех.

Исходя из этих трех значений высчитывается координата отсчета, ширина и высота контейнера.

Далее происходит проверка на пересечения с другими контейнерами. Производится обход дерева в ширину. Если контейнеры пересекаются и имеют общего родителя - новые координаты контейнеров-потомков высчитываются относительно центра контейнера – родителя с силой отталкивания repulsiveForce (рис. 5).

Если контейнеры являются потомками разных родителей – производится поиск общего предка выше. Когда общий предок найден, его узлы-потомки разводятся с двойной отталкивающей силой от центра родительского блока (repulsiveForce * 2) (рис. 6).

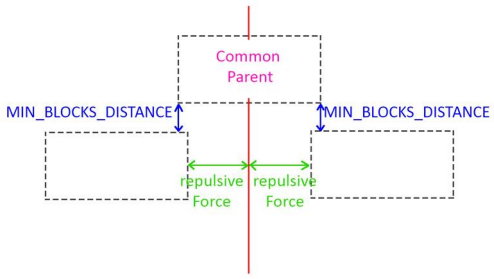

Рис. 5. Вычисление координат контейнеров с общим родителем

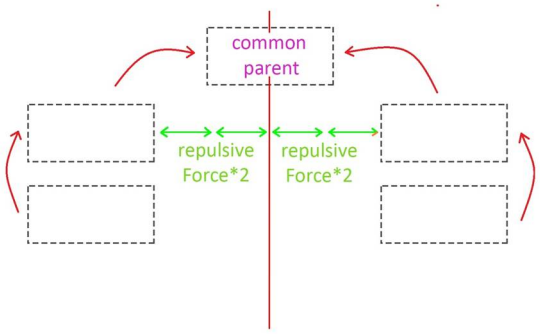

Рис. 6. Вычисление координат контейнеров без общего родителя

Далее перерисовываются все его потомки, проводится проверка на пересечения.

ИНСТРУМЕНТЫ ПРОГРАММНОЙ РЕАЛИЗАЦИИ ВИЗУЛЬНОГО РЕДАКТОРА СЛОЖНЫХ-БЛОК СХЕМ

Описанный механизм пространственного размещения символов-блоков для сложных блок-схем был реализован в визуальном редакторе в виде интегрированной среды обработки. Редактор представляет собой веб-приложение, реализованное на языках TypeScript и JavaScript с использованием библиотек React, Redux и многих других, позволяющих сделать представление элементов приятным, процесс создания приложения – быстрым.

РЕАЛИЗАЦИЯ МЕХАНИЗМА ВИЗУАЛИЗАЦИИ СЛОЖНЫХ БЛОК-СХЕМ

Визуальный редактор реализован в виде веб-приложения, внешний вид клиентской части которого представлен на рисунке 7.

Сокрытие деталей реализации производится по клику на контейнер. Отображение скрытого контейнера производится по клику на родительском блоке.

ЗАКЛЮЧЕНИЕ

В данной работе приведены результаты исследования и разработки редактора блок-схем алгоритмов. Этот редактор представляет собой один из компонентов проектируемой многоязыковой визуально-графической системы программирования. При ее разработке приходится решать ряд сложных задач, в числе которых – задача автоматического

размещения текущего представления элементов (блоков, контейнеров) и связей между ними, в окне, наложенном на потенциально значительно большую по размерам блок-схему проектируемого алгоритма. Способ представления (свернутый или развернутый) каждого контейнера блок-схемы, возможно, содержащего десятки или сотни вложенных в него элементов, выбирается пользователем.

Предложен и описан способ пространственного размещения элементов/контейнеров сложных алгоритмов в автоматическом режиме с учетом необходимости реализации всех потенциально возможных операций пользователя по модификации проектируемой блок-схемы. Рассмотрены необходимые для реализации этого подхода внутренние структуры данных визуально-графического редактора и некоторые наиболее важные алгоритмы расчета/пересчета координат блоков/контейнеров и связей между ними в процессе автоматического размещения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Grzegorz Polaków, Mieczyslaw Metzger. Dynamically Generated Block Diagrams as a Visualisation Method // IFAC Proceedings Volumes. – 2009. - Volume 42, Issue 13. - P. 109-118

2. Род Стивенс. Алгоритмы. – М.: Издательство «Э», 2017. – 215-295 С.

3. Ландовский В. В. Структуры данных: учеб. Пособие – Новосибирск: Изд-во НГТУ, 2016. - 40 С.

4. Поляков Р. R* - tree или индексация геопространственных данных. [Электронный ресурс]: Хабр – 2014. – Режим доступа: https://habr.com/ru/post/224965.

Евдокимова Ольга Алексеевна – студент, магистрант, Новосибирский государственный технический университет, тел. (383)3460492, e-mail: evdokimova_olga_15@mail.ru.

Вохмин Александр Андреевич – студент, магистрант, Новосибирский государственный технический университет, тел. (383)3460492, e-mail: alexandervokhmin@gmail.com.

Малявко Александр Антонович – кандидат технических наук, доцент кафедры ВТ, Новосибирский государственный технический университет, тел. (383)3460492, e-mail: a.malyavko@corp.nstu.ru.

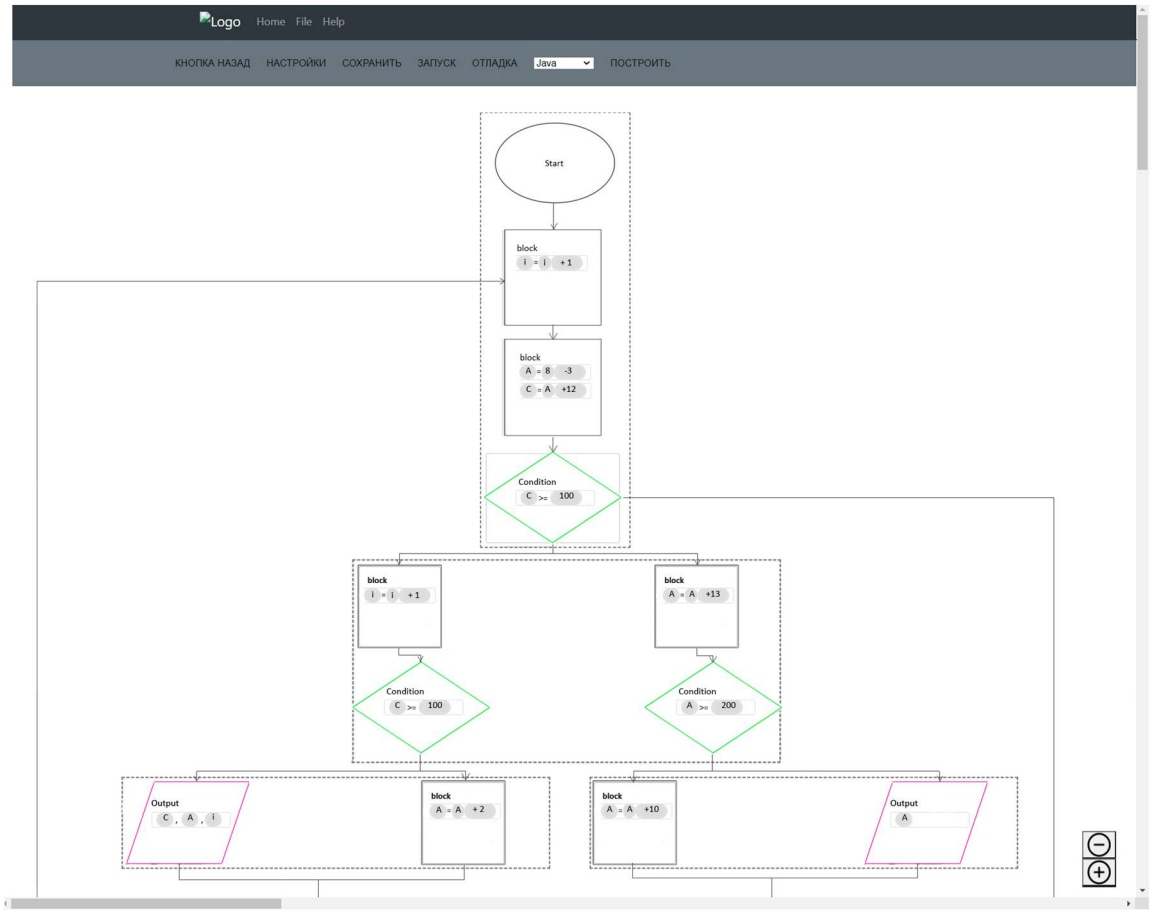

Рис. 7. Общий вид программы с использованием разработанного алгоритма пространственного размещения

VISUAL-GRAPHIC PROGRAMMING SYSTEM BASED ON DEVELOPMENT OF ALGORITHM BLOCK SCHEMES. SPATIAL PLACEMENT OF SYMBOLS-BLOCKS OF COMPLEX ALGORITHMS

O.A. Evdokimova, A.A. Vokhmin, A.A. Malyavko
Novosibirsk State Technical University, Novosibirsk

Abstract – The article presents the results of research and development of the algorithm block diagram editor. The editor is one of the components of the projected multilingual visual-graphic programming system. When developing it, it is necessary to solve a number of complex problems, including the problem of automatic placement of the representation of elements (blocks, containers) and links between them in a window superimposed on a large-sized block diagram of an object. predictable algorithm. The presentation method (collapsed or expanded) of each container of the flowchart, which can contain tens or hundreds of nested elements, can be selected by the user at each stage of the algorithm development.

A method of spatial arrangement of elements / containers of complex algorithms in automatic mode is proposed and described, taking into account the implementation of all possible user actions to modify the projected block diagram. The necessary approaches for their implementation are the internal data structures of the visual-graphic editor and some of the most important algorithms for calculating / recalculating the coordinates of blocks / containers and the links between them in the process of automatic placement.

Index terms: visual editor, structural diagram, algorithm, spatial arrangement, calculation of coordinates.

REFERENCES

1. Grzegorz Polaków, Mieczyslaw Metzger. Dynamically Generated Block Diagrams as a Visualisation Method // IFAC Proceedings Volumes. – 2009. - V 42, I 13. - P. 109-118.

2. Rod Stevens. Algorithms. "E". Moscow, Russia: 2017.

3. Landovskiy V.V. Data structures: textbook. Manual. Novosibirsk, Russia: Publishing house of NSTU, 2016.

4. Polyakov R. R * - tree or indexing of geospatial data. Habr, accessed April 30, 2021, https://habr.com/ru/post/224965.

Evdokimova Olga Alekseevna – student, Novosibirsk State Technical University, (383)3460492, e-mail: evdokimova_olga_15@mail.ru.

Vokhmin Alexander Andreevich – student, Novosibirsk State Technical University, (383)3460492, e-mail: alexandervokhmin@gmail.com.

Malyavko Alexander Antonovich – Candidate of Technical Sciences, Associate Professor of the Department of Computing Technologies, Novosibirsk State Technical University, (383)3460492, e-mail: a.malyavko@corp.nstu.ru.