УДК 004.4'236
Doi: 10.31772/2712-8970-2023-24-1-8-17

Для цитирования: Касьянов В. Н. Методы и средства визуализации информации на основе атрибутированных иерархических графов с портами // Сибирский аэрокосмический журнал. 2023. Т. 24, № 1. С. 8–17. Doi: 10.31772/2712-8970-2023-24-1-8-17.

For citation: Kasyanov V. N. [Methods and tools for information visualization on the basis of attributed hierarchical graphs with ports]. *Siberian Aerospace Journal.* 2023, Vol. 24, No. 1, P. 8–17. Doi: 10.31772/2712-8970-2023-24-1-8-17.

Методы и средства визуализации информации на основе атрибутированных иерархических графов с портами

В. Н. Касьянов

Институт систем информатики имени А. П. Ершова СО РАН
Российская Федерация, 630090, Новосибирск, просп. Академика Лаврентьева, 6
E-mail: kvn@iis.nsk.su

В настоящее время визуализация графовых моделей является неотъемлемой частью обработки сложной информации о структуре объектов, систем и процессов во многих приложениях в науке и технике и на рынке широко представлены наукоемкие программные продукты, использующие методы визуализации информации на основе графовых моделей. Поскольку информация, которую желательно визуализировать, постоянно увеличивается и усложняется, возникает все больше ситуаций, в которых классические графовые модели перестают быть адекватными. Требуются и возникают более мощные теоретико-графовые формализмы для представления информационных моделей, обладающих иерархической структурой, поскольку иерархичность является основой многочисленных методов визуальной обработки сложных больших данных в различных областях применениям. Одним из таких формализмов являются так называемые иерархические графы. Этот формализм позволяет выделить в исходном классическом графе множество таких его частей (так называемых фрагментов), что все элементы каждого фрагмента заслуживают отдельного совместного рассмотрения, а все фрагменты выделенного множества образуют иерархию по вложенности. В Институте систем информатики им. А. П. Ершова СО РАН была создана система визуализации Visual Graph, которая основана на иерархических графах и позволяет исследовать сложные структурированные большие данные через их визуальные представления. Во многих приложениях объекты, моделируемые вершинами графа, являются сложными и содержат непересекающиеся логические части (так называемые порты), через которые эти объекты находятся во взаимосвязи, моделируемой ребрами. В статье введен формализм атрибутированных иерархических графов с портами и рассмотрены новые возможности системы Visual Graph по визуализации структурированных данных большого размера на основе атрибутированных иерархических графов с портами.

Ключевые слова: атрибутированный иерархический граф, визуализация информации, графовая модель, порт, система визуализации.

Methods and tools for information visualization on the basis of attributed hierarchical graphs with ports

V. N. Kasyanov

A. P. Ershov Institute of Informatics Systems SB RAS
6, Acad. Lavrentjev pr., Novosibirsk, 630090, Russian Federation
E-mail: kvn@iis.nsk.su

At present visualization of graph models is an inherent part of the processing of complex information about the structure of objects, systems and processes in many applications in science and technology, and at the market there are widely presented science-intensive software products, using the information visualization on the basis of graph models. Since the information that it is desirable to visualize is constantly growing and becoming more complex, more and more situations arise in which classical graph models cease to be adequate. More powerful graph-theoretic formalisms are required and appear to represent information models with a hierarchical structure, since hierarchy is the basis of numerous methods for visual processing of complex big data in various fields of application. One of these formalisms is the so-called hierarchical graphs. This formalism allows selecting in the given classical graph a set of such its parts (so-called fragments) that all elements of each selected fragment deserve separate joint consideration, and all fragments of the selected set form a nesting hierarchy. At the A. P. Ershov Institute of Informatics Systems constructed the Visual Graph visualization system, which is based on hierarchical graphs and allows exploring complex structured big data through their visual representations. In many applications, objects modeled by graph vertices are complex and contain non-intersecting logical parts (so-called ports) through which these objects are in a relationship modeled by arcs. In the paper the formalism of attributed hierarchical graphs with ports is introduced and new possibilities of the Visual Graph system for visualization of large structured data based on attributed hierarchical graphs with ports are considered.

Keywords: attributed hierarchical graph, data visualization, graph model, port, visualization system.

Введение

Визуализация структурированных или реляционных данных на основе графовых моделей имеет множество сфер применения как в реальных, так и в теоретических областях [1–3]. Среди них – физические сети связи и электрические сети, с одной стороны, и структуры данных компилятора и диаграммы изменения состояний – с другой. Поэтому в настоящее время на рынке широко представлены наукоемкие программные продукты, использующие методы визуализации информации на основе графовых моделей, такие как Cytoscape [4], Higres [5], Gephi [6], Graphviz [7], Tulip [8], yEd [9] и многие другие.

Поскольку информация, которую желательно визуализировать, постоянно увеличивается и усложняется, возникает все больше ситуаций, в которых классические графовые модели перестают быть адекватными. Требуются и возникают более мощные теоретико-графовые формализмы для представления информационных моделей, обладающих иерархической структурой, поскольку иерархичность является основой многочисленных методов визуальной обработки сложных больших данных в различных областях применения вычислительных систем [3; 10; 11]. Одним из таких формализмов являются так называемые иерархические графы и графовые модели [3]. Этот формализм позволяет выделить в исходном графе множество таких его частей (так называемых фрагментов), что все элементы каждого фрагмента заслуживают отдельного совместного рассмотрения, а все фрагменты выделенного множества образуют иерархию по вложенности. Такая иерархия фрагментов обеспечивает хорошие возможности для борьбы со сложностью при визуализации структурированных данных в тех приложениях, которые обязаны иметь дело с большими объемами сложных структурированных данных. Она формирует

основу для естественных методов «абстракции» и «редукции», которые могут применяться исследователями для уменьшения визуальной сложности большого графа. В Институте систем информатики им. А. П. Ершова СО РАН по заказу компании Intel была создана система визуализации Visual Graph, которая основана на иерархических графовых моделях и позволяет исследовать сложные структурированные большие данные, возникающие в компиляторах и других системах конструирования программ, через их визуальные представления [12].

Во многих приложениях объекты, моделируемые вершинами графа, являются сложными и могут содержать по несколько разных логических частей, через которые эти объекты находятся во взаимосвязи, моделируемой ребрами. Например, в графе железнодорожных дорог страны (или некоторой другой территории) населенные пункты, моделируемые вершинами графа, могут соединяться железнодорожными путями, заходящими в разные железнодорожные вокзалы этих населенных пунктов. А при представлении потока данных между операторами программы в виде так называемого информационного графа у операторов программы, моделируемых вершинами графа, рассматриваются разные их операнды (так называемые информационные входы – разные те места, где данные используются в качестве аргументов операторов, и информационные выходы – разные те места, где данные возникают в качестве результатов операторов), через которые и происходит при исполнении программы обмен данными между операторами (от выходов к входам), и поэтому информационные связи между операторами (вершинами информационного графа) обычно представлены ориентированными ребрами (дугами), которые соединяют соответствующие операнды операторов [3]. При представлении графов с вершинами, моделирующими сложные объекты, в существующих форматах описания графов (например, стандартный формат описания графов GraphML [13]) эти разные логические части сложных объектов обычно выражаются с помощью так называемых портов вершин, которые при изображении графа могут представляться разными точками (или разными непересекающимися частями) изображений вершин, в которых соответствующие вершины соединяются с инцидентными им ребрами.

В данной статье введен формализм атрибутированных иерархических графов с портами, описаны способы изображения таких графов на плоскости и кратко представлены новые возможности системы Visual Graph по визуализации структурированных данных большого размера на основе атрибутированных иерархических графов с портами. Статья является расширенной версией доклада, прочитанного автором на III Сибирском научном семинаре Data Analysis Technologies with Applications (SibDATA-2022) [14].

Формализм атрибутированных иерархических графов с портами и способы изображения таких графов на плоскости

Напомним некоторые термины и обозначения из [3].

Пусть G – граф некоторого типа, например, G может быть неориентированным или ориентированным графом. Граф G определяется двумя конечными множествами V и E, где элементы V – вершины графа G, а элементы E – ориентированные (или неориентированные) ребра графа G. G – тривиальный граф, если $|V|=1$ и $|E|=0$.

Граф C называется фрагментом графа G и обозначается $C \subseteq G$, если C – часть графа G, т. е. состоит только из элементов (вершин и ребер) графа G.

Множество фрагментов F называется иерархией вложенных фрагментов графа G, если

(1) F содержит граф G и

(2) $C_1 \subseteq C_2$, $C_2 \subseteq C_1$ или пусто $C_1 \cap C_2$ для любых $C_1, C_2 \in F$.

Для любых различных $C_1, C_2 \in F$ фрагмент C_1 непосредственно вложен в C_2, если $C_1 \subseteq C_2$ и не существует такого $C_3 \in F$, отличного от C_1 и C_2, что $C_1 \subseteq C_3 \subseteq C_2$.

Фрагмент $C \in F$ – элементарный, если F не содержит фрагментов, непосредственно вложенных в C.

Иерархический граф $H=(G,T)$ состоит из графа *G* и корневого дерева *T*, которое представляет отношение непосредственной вложенности между элементами некоторой иерархии *F* вложенных фрагментов *G*. *G* называется основным графом *H*. *T* называется деревом вложенности *H*.

Иерархический граф $H=(G,T)$ называется простым, если все его фрагменты являются порожденными подграфами графа *G*.

Нетрудно увидеть, что каждый кластерный граф может быть рассмотрен как простой иерархический граф $H=(G,T)$, в котором *G* – неориентированный граф, а листья дерева *T* – тривиальные подграфы графа *G*.

Пример простого иерархического графа $H=(G,T)$ приведен на рис. 1, на котором ребра основного графа *G* изображены сплошными линиями, а ребра дерева вложенности *T* – пунктирными. Данный простой иерархический граф *H* содержит шесть тривиальных фрагментов и два нетривиальных фрагмента: *G* и $C=\{1,\ 2,\ 3.5\}$.

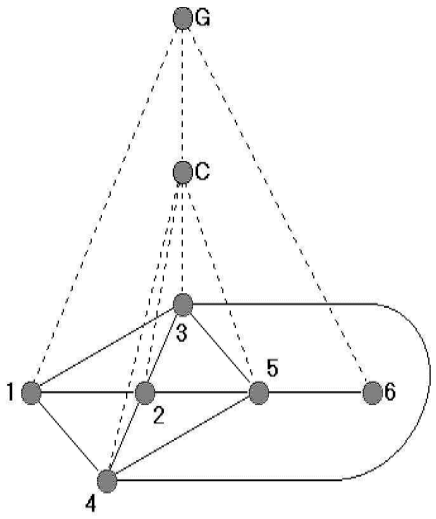

Рис. 1. Простой иерархический граф

Fig. 1. A simple hierarchical graph

Определим иерархические графы с портами как подкласс иерархических графов следующим образом.

Пусть $H=(G,T)$ – иерархический граф и пусть $P\subseteq V$ – некоторое выделенное подмножество вершин графа *G*. Будем называть портами фрагмента *C* все те его вершины из *P*, которые не принадлежат ни одному вложенному в него фрагменту из *F*.

Нетрудно увидеть, что в любом иерархическом графе с портами $H=(G,T)$ каждая вершина из *P* является портом некоторого фрагмента $C\in F$ и только одного его фрагмента.

Таким образом, множество всех вершин графа $H=(G,T)$ распадается на три попарно непересекающихся множества:

(1) множество всех портов *P*;

(2) множество простых вершин – всех тех вершин его основного графа *G*, которые не являются портами фрагментов из *F*;

(3) множество всех тех вершин дерева вложенности *T*, которые не являются тривиальными фрагментами основного графа.

Например, информационный граф программы может быть рассмотрен как такой иерархический граф с портами $H=(G,T)$, в котором *G* – ориентированный граф с $P=V$, а каждый фрагмент $C\in F$, отличный от *G*, является элементарным пустым фрагментом, множество портов которого распадается на два таких непересекающихся подмножества $In(C)$ и $Out(C)$, что нет дуг, исходящих из портов $In(C)$ или заходящих в порты $Out(C)$. При таком рассмотрении элементарные фрагменты $C\in F$ моделируют операторы программы, $In(C)$ и $Out(C)$ – множества информационных входов и выходов соответствующих операторов, а дуги основного графа *G* – информационные связи между соответствующими выходами и входами операторов.

Изображение (или укладка) иерархического графа с портами $H=(G,T)$ является таким представлением элементов *H* на плоскости, что выполняются следующие свойства:

1. Каждая вершина графа *H* представлена некоторой замкнутой областью (например, кругом или прямоугольником). Область определяется ее границей (простой замкнутой кривой на плоскости), которая делит оставшуюся часть плоскости на две части: внутреннюю грань и внешнюю грань. Другими словами область, изображающая вершину графа *H*, состоит из ее границы и ее внутренней грани.

2. Для любых $C_1, C_2 \in F$ пересечение областей фрагментов C_1 и C_2 пусто тогда и только тогда, когда пусто $C_1 \cap C_2$, а область каждого фрагмента $C \in F$ включает в себя области всех вложенных в него фрагментов и области всех ее портов и простых вершин.

3. Область любого порта и любой простой вершины любого фрагмента $C \in F$ не содержит точек областей других портов и простых вершин фрагмента *C* и точек областей тех фрагментов, которые вложены во фрагмент *C*.

4. Каждое ребро графа *G* представлено простой кривой (со стрелкой, если это ребро ориентировано), соединяющей две точки, принадлежащие границам тех двух областей, которые изображают инцидентные данному ребру вершины.

5. Все ребра любого фрагмента $C \in F$ расположены внутри области фрагмента *C*.

6. Если простая вершина, порт или ребро *h* графа *H* не принадлежит некоторому его фрагменту $C \in F$, то область фрагмента *C* не содержит точек области простой вершины или порта *h* и не содержит точек кривой, изображающей такое ребро *h*, которое соединяет вершины, не принадлежащие фрагменту *C*.

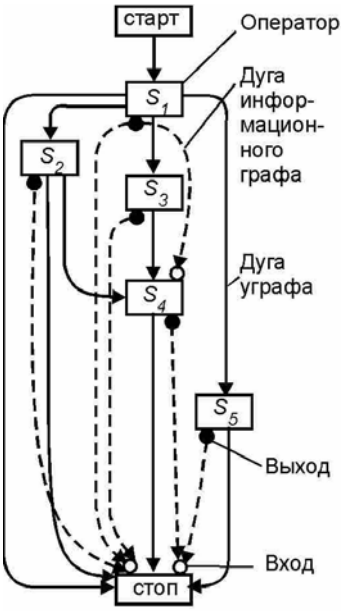

Рис. 2. Р-схема

Fig. 2. P-diagram

В качестве примера рассмотрим использование введенного формализма для изображения операторной схемы над распределенной памятью (или Р-схемы) [3]. В операторной схеме данного типа управляющий граф (уграф) программы, вершины которого соответствуют операторам программы, а дуги представляют управляющие связи (возможные передачи управления) между ними, как бы дополнен информационным графом, представляющим информационные связи между соответствующими выходами и входами операторов. На рис 2 приведен пример изображения конкретной схемы над распределенной памятью. Здесь, как и обычно, операторы программы изображены в виде прямоугольников, снабженных кругами, изображающими операнды операторов (входы оператора располагаются сверху соответствующего прямоугольника, а выходы снизу). Управляющие связи изображены сплошными линиями (со стрелками), соединяющими операторы, а информационные связи – пунктирными линиями (со стрелками), соединяющими выходы операторов с входами.

Каждая Р-схема может быть представлена в виде такого иерархического графа с портами $H = (G, T)$, в котором любой фрагмент $C \in F$, отличный от основного графа *G*, является элементарным пустым фрагментом, состоящим из одной основной вершины $q \in V \setminus P$ и возможно пустого множества портов $p \in P$, которое распадается на два непересекающихся подмножества $In(C)$ и $Out(C)$. При таком представлении простые вершины q элементарных фрагментов $C \in F$ моделируют операторы Р-схемы, а порты p из множеств $In(C)$ и $Out(C)$ моделируют их информационные входы и выходы. Дуги основного графа *G*, соединяющие вершины $q \in V \setminus P$, – это управляющие связи, а дуги основного графа *G*, соединяющие порты $p \in P$, – это информационные связи между соответствующими выходами и входами операторов. Если изображать каждый элементарный фрагмент такого иерархического графа в виде фигуры, образованной прямоугольником, изображающим его вершину, и примыкающими к нему кругами, изображающими его порты (входы сверху прямоугольника, а выходы снизу), а каждую дугу основного графа рисовать либо сплошной линией со стрелкой, если это управляющая связь, либо пунктирной линией со стрелкой в случае информационной связи, то можно получить стандартное изображение Р-схемы, приведенное на рис. 2.

Пусть задано множество объектов W, называемых атрибутами, и пусть каждому элементу $w \in W$ сопоставлено в соответствие множество объектов $B(w)$, называемых возможными значениями атрибута w. Например, в качестве $B(w)$ могут использоваться определенные множества чисел, символов или строк (цепочек символов). Пусть M обозначает множество всех пар (w,v), образованных из атрибутов $w \in W$ и их значений $v \in B(w)$.

Атрибутированный иерархический граф с портами – это пара (H,L), где H – иерархический граф с портами, а L – функция атрибутирования, сопоставляющая каждому его элементу h некоторое подмножество $L(h) \subseteq M$.

При изображении атрибутированного иерархического графа с портами атрибуты и их значения для элементов иерархического графа могут либо выражаться неявно через определенные свойства способа представления этих элементов (например, через геометрическую форму области вершины, ее размеры и т. д.), либо изображаться явно в виде определенного вида пометок соответствующих элементов. Например, явное представление атрибутов может определять место и вид изображений атрибутов в виде текстов внутри областей вершин или рядов с линиями, изображающими ребра.

Новые возможности системы Visual Graph по визуализации структурированных данных большого размера на основе атрибутированных иерархических графов с портами

Система Visual Graph [12] предназначена главным образом для визуального и структурного анализа структурированных данных, возникающих при работе компилятора (или другой системы конструирования программ), на основе атрибутированных иерархических графов достаточно большого размера, а теперь еще и с портами. Предполагается следующий сценарий использования системы (рис. 3). Сначала компилятор (сам либо с помощью вспомогательной программы) переводит графовую модель из ее внутреннего представления в файл одного из поддерживаемых системой Visual Graph форматов, как правило, GraphML-файл. После этого система Visual Graph сможет прочитать эту графовую модель из файла, визуализировать ее и предоставить пользователю средства навигации по ней и ее анализа. В отличие от зарубежных аналогов (и другой нашей системы Higres), система Visual Graph поддерживает обработку произвольных атрибутированных иерархических графов достаточно большого размера, ориентирована на пошаговое построение многооконного изображения графовой модели, состоящего из укрупненных изображений интересных пользователю фрагментов модели, и предоставляет богатые возможности для навигации по графовой модели и ее структурного анализа, работы с атрибутами ее элементов, а также простого расширения и настройки системы на нужды конкретного пользователя.

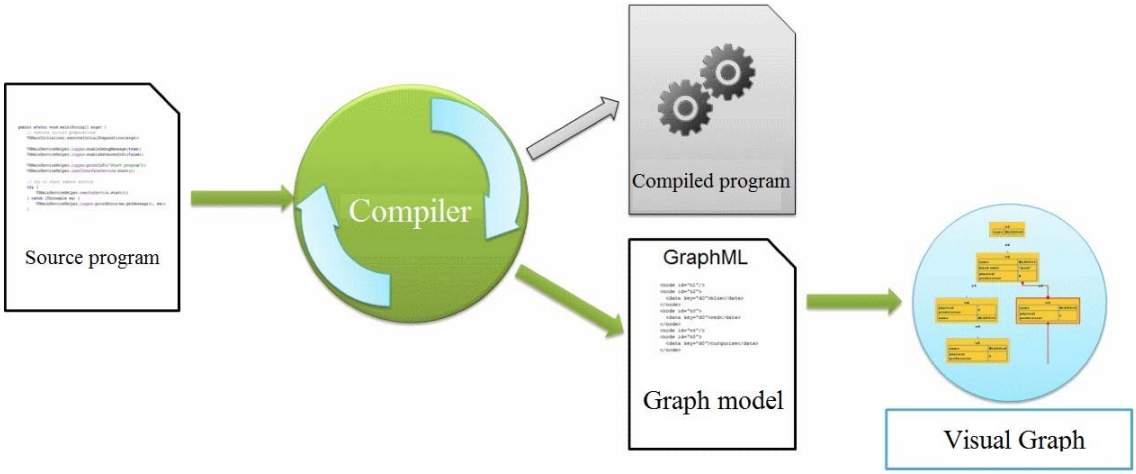

Рис. 3. Сценарий использования системы Visual Graph

Fig. 3. The scenario of using the Visual Graph system

Изначально система поддерживала только один формат для представления графов – это стандартный язык описания графов GraphML. В текущей же версии системы добавлена поддержка таких форматов, как DOT [15] и GML [16], поскольку некоторые весьма популярные компиляторы и системы визуализации ориентированы на работу с этими форматами. Добавлена также возможность экспорта системой частей графа в GraphML-формат для их дальнейшей обработки.

В текущей версии системы изменился набор встроенных алгоритмов укладки графов за счет его расширения двумя алгоритмами: алгоритмом циркулярной укладки атрибутированных иерархических графов с портами [17], а также алгоритмом поуровневой укладки атрибутированных иерархических графов с портами, представляющими внутреннее представление функциональных программ [18]. При этом, библиотека JGraph [19], которая изначально использовалась в системе для отображения и укладки графов, в текущей версии системы заменена на специально разработанный модуль для отображения графовых моделей. Дело в том, что библиотека JGraph имеет ряд ограничений, которые затрудняли реализацию работы с атрибутированными иерархическими графами с портами в полном объеме. Основными трудностями ее использования для реализации новых встроенных алгоритмов укладки стали проблемы, связанные с существенно возросшим за счет портов количеством элементов графов, а также с требуемыми изменениями способов отображения портов и фрагментов.

Произошли также изменения в пользовательском интерфейсе системы, который по-прежнему включает рабочий стол, навигационную панель, а также атрибутную панель (рис. 4).

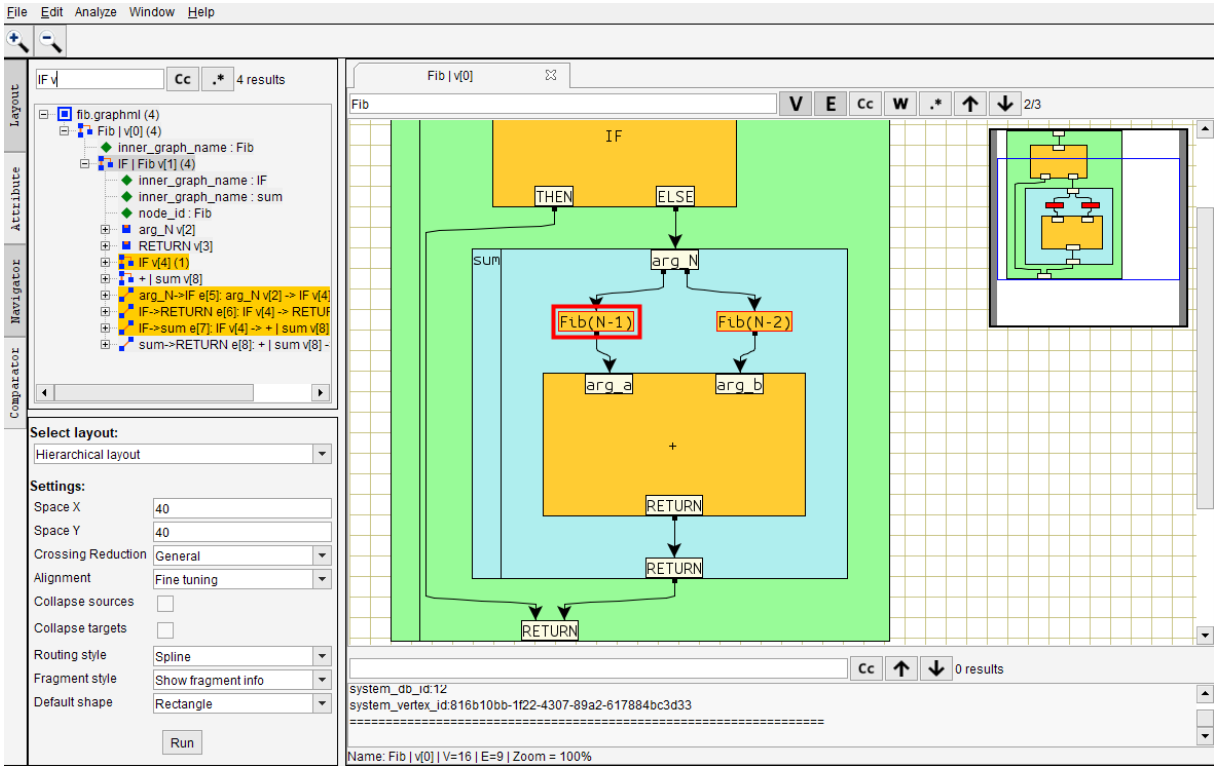

Рис. 4. Система Visual Graph

Fig. 4. Visual Graph System

Рабочий стол теперь состоит из набора вкладок, открываемых пользователем для визуального и структурного анализа выбранных фрагментов иерархической графовой модели в виде их изображений на плоскости, построенных с помощью встроенных в систему алгоритмов укладки. К средствам структурного анализа относятся различные встроенные алгоритмы обработки графовой модели, которые помогают пользователю выделять и визуализировать нужную ему

информацию в изображениях графов. К ним относятся, например, такие средства, как поиск кратчайших путей, циклов или обязательных предшественников в графе, а также поиск максимального общего подграфа двух графов. Каждый из встроенных алгоритмов укладки и алгоритмов структурного анализа имеет свой набор параметров, позволяющих управлять как видом полученного в результате его работы изображения графовой модели, так и процессом его построения. Кроме того, для улучшения изображения, полученного автоматически, пользователь может вручную менять форму вершин и ребер, отображаемые атрибуты, масштаб видимой области и многое другое, а также теперь может использовать сетку при осуществлении этих изменений. Каждая вкладка имеет свой фильтр (поисковую строку), используя который можно найти и выделить все те элементы у видимой части графовой модели, текстовые атрибуты которых удовлетворяют регулярному выражению, введенному пользователем в поисковую строку. Расширены возможности фильтра по его настройке на то, какие элементы нужно искать. Так, например, можно осуществлять поиск только среди вершин или ребер. Мини-карта позволяет обозревать весь граф целиком вместе с выделенной той его частью, которая видна в текущей вкладке, а также предоставляет пользователю возможность перемещать и масштабировать видимую часть графа. Сама мини-карта была перенесена с панели инструментов, где она размещалась первоначально, в правую верхнюю часть текущей вкладки. Также была добавлена возможность отключения показа мини-карты; что позволяет пользователю, если необходимо, увеличить по вертикали размер той области текущей вкладки, которая используется для рассмотрения построенного изображения видимого фрагмента графа.

Навигационная панель – это инструмент для визуализации всех частей графов, с которыми в данный момент работает пользователь, в виде изображения с помощью отступов деревьев вложенности фрагментов этих графов. Поддерживаются операции свертки и развертки изображений поддеревьев вложенности, а также выделения интересных пользователю фрагментов и их открытия в новых вкладках. Для быстрого поиска по деревьям был доработан фильтр (строка поиска), который позволяет пользователю без труда найти интересующие его фрагменты, используя регулярные выражения. Отличия от первоначальной версии заключаются в предоставлении большей гибкости при задании условий для поиска, а также в улучшении работы с атрибутами, содержащими большие объемы данных.

Атрибутная панель – инструмент, который позволяет управлять визуализацией атрибутов для выбранных вершин и ребер в текущей вкладке. Для этого пользователю необходимо выделить у графа из текущей вкладке те вершины и ребра, которые ему интересны, после чего отметить в атрибутной панели галочками те атрибуты, которые он хочет визуализировать у этих элементов. Так же с помощью данного инструмента можно задать набор атрибутов, которые будут визуализироваться для элементов фрагментов, открытых в новой вкладке. Панель с атрибутами была перенесена из нижней секции пользовательского интерфейса в секцию слева, что позволило увеличить просматриваемую область графового изображения. Значения атрибутов были убраны с панели атрибутов и перенесены в область графового изображения, и теперь, когда пользователь выделяет элементы графового изображения, автоматически появляется информация о выбранных элементах. Данная информация представлена в виде сплошного текста, в котором можно осуществлять поиск, а также копировать части текста и переносить их в другие сторонние утилиты. При желании можно скрыть данное меню.

Заключение

В статье введен формализм атрибутированных иерархических графов с портами, а также описаны способы изображения таких графов на плоскости и новые возможности системы Visual Graph по поддержке на основе этого формализма визуализации структурированных данных большого размера, возникающих в компиляторах и других системах конструирования программ. В дальнейшем мы планируем развитие системы Visual Graph, главным образом, путем расширения набора встроенных алгоритмов укладки и структурного анализа атрибутированных

иерархических графов с портами различными алгоритмами, которые интересны не только для систем конструирования программ, но и других приложений, а также путем придания системе возможностей предоставления соответствующего веб-сервиса.

Библиографические ссылки

1. Graph Drawing: Algorithms for Visualization of Graphs / Di Battista G., Eades P., Tamassia R. et al. Prentice Hall, 1999. 379 p.

2. Herman I., Melançon G., Marshall M. S. Graph visualization and navigation in information visualization: a survey // IEEE Transactions on Visualization and Computer Graphics. 2000. Vol. 6. P. 24–43.

3. Касьянов В. Н., Евстигнеев В. А. Графы в программировании: обработка, визуализация и применение. СПб.: БХВ-Петербург, 2003. 1104 с.

4. Cytoscape [Электронный ресурс]. URL: https://cytoscape.org

5. Lisitsyn I. A., Kasyanov V. N. Higres – visualization system for clustered graphs and graph algorithms // Lecture Notes in Computer Science. 1999. Vol. 1731. P. 82–89.

6. Gephi [Электронный ресурс]. URL: https://gephi.org.

7. Graphviz [Электронный ресурс]. URL: https://graphviz.org.

8. Tulip [Электронный ресурс]. URL: https://tulip.labri.fr/TulipDrupal.

9. yEd [Электронный ресурс]. URL: https://www.yworks.com/products/yed.

10. Feng Q. W., Cohen R .F., Eades P. Planarity for clustered graphs // Lecture Notes in Computer Science. 1995. Vol. 979. P. 213–226.

11. Sugiyama K. Misue K. Visualization of structured digraphs // IEEE Transactions on Systems, Man and Cybernetics. 1999. Vol. 21. No. 4. P. 876–892.

12. Касьянов В. Н., Золотухин Т. А. Visual Graph – система для визуализации сложно структурированной информации большого объема на основе графовых моделей // Научная визуализация. 2015. Т. 7, № 4. С. 44–59.

13. GraphML progress report: structural layer proposal / U. Brandes, M. Eiglsperger, I. Herman et al. // Lecture Notes in Computer Science. 2002. Vol. 2265. P. 501–512.

14. Касьянов В. Н. Визуализация структурированных данных на основе атрибутированных иерархических графов с портами // III Сиб. науч. семинар Data Analysis Technologies with Applications (SibDATA-2022). Красноярск : ИВМ СО РАН, 2022. С. 9–10.

15. DOT [Электронный ресурс]. URL: http://www.graphviz.org/doc/info/lang.html.

16. GML [Электронный ресурс]. URL: http://openmis.ru/doc/clang/gml-tr.html.

17. Kasyanov V. N., Merculov A. M., Zolotuhin T. A. A circular layout algorithm for attributed hierarchical graphs with ports // Journal of Physics: Conference Series. 2021. Vol. 2099. P. 012051.

18. Касьянов В. Н., Золотухин Т. А., Гордеев Д. С. Методы и алгоритмы визуализации графовых представлений функциональных программ // Программирование. 2019. № 4. С. 19–27.

19. JGraph [Электронный ресурс]. URL: http://dev.cs.ovgu.de/java/jgraph/tutorial/t1.html.

References

1. Di Battista G., Eades P., Tamassia R. et al. Graph Drawing: Algorithms for Visualization of Graphs. Prentice Hall, 1999, 379 p.

2. Herman I., Melançon G., Marshall M. S. Graph visualization and navigation in information visualization: a survey. *IEEE Transactions on Visualization and Computer Graphics*. 2000, Vol. 6, P. 24–43.

3. Kasyanov V. N., Evstigneev V. A. *Grafy v programmirovanii: obrabotka, vizualizatsiya i primenenie* [Graphs in Programming: Processing, Visualization and Application]. St. Petersburg: BHV-Petersburg, 2003, 1104 p.

4. Cytoscape. Available at: https://cytoscape.org.

5. Lisitsyn I. A., Kasyanov V. N. Higres – visualization system for clustered graphs and graph algorithms. *Lecture Notes in Computer Science*. 1999, Vol. 1731, P. 82–89.

6. Gephi. Available at: https://gephi.org.

7. Graphviz. Available at: https://graphviz.org.

8. Tulip. Available at: https://tulip.labri.fr/TulipDrupal.

9. yEd homepage. Available at: https://www.yworks.com/products/yed.

10. Feng Q. W., Cohen R. F., Eades P. Planarity for clustered graphs. *Lecture Notes in Computer Science*. 1995, Vol. 979, P. 213–226.

11. Sugiyama K. Misue K. Visualization of structured digraphs. *IEEE Transactions on Systems, Man and Cybernetics*. 1999, Vol. 21, No. 4, P. 876–892.

12. Kasyanov V. N., Zolotuhin T. A. [Visual Graph – a system for visualization of big size complex structural information on the base of graph models]. *Scientific Visualization*. 2015, Vol. 7, No. 4, P. 44–59. (In Russ.).

13. Brandes U., Eiglsperger M., Herman I. et al. GraphML progress report: structural layer proposal. *Lecture Notes in Computer Science*. 2002, Vol. 2265, P. 501–512.

14. Kasyanov V. N. [Visualization of structured data based on attributed hierarchical graphs with ports]. *III Siberian Scientific Workshop on Data Analysis Technologies with Applications (SibDATA-2022)*. Krasnoyarsk, ICM SB RAS Publ., P. 9–10.

15. DOT. Available at: http://www.graphviz.org/doc/info/lang.html.

16. GML. Available at: http://openmis.ru/doc/clang/gml-tr.html.

17. Kasyanov V. N., Merculov A. M., Zolotuhin T.A. A circular layout algorithm for attributed hierarchical graphs with ports. *Journal of Physics: Conference Series*. 2021, Vol. 2099, P. 012051.

18. Kasyanov V. N., Zolotuhin T. A., Gordeev D. S. Visualization methods and algorithms for graph representation of functional programs. *Programming and Computer Software*. 2019, Vol. 45, No. 4, P. 156–162.

19. JGraph. Available at: http://dev.cs.ovgu.de/java/jgraph/tutorial/t1.html.

Касьянов Виктор Николаевич – доктор физико-математических наук, профессор, главный научный сотрудник; Институт систем информатики имени А. П. Ершова СО РАН. E-mail: kvn@iis.nsk.su.

Kasyanov Victor Nikolaevich – Dr. Sc., Professor, Chief Researcher; A. P. Ershov Institute of Informatics Systems. E-mail: kvn@iis.nsk.su.