И.А. Лисицын

ОРГАНИЗАЦИЯ ГРАФИЧЕСКОГО ВЫВОДА В СИСТЕМЕ ВИЗУАЛИЗАЦИИ ИЕРАРХИЧЕСКИХ ГРАФОВЫХ МОДЕЛЕЙ[1]

1. ВВЕДЕНИЕ

Графы широко применяются в различных областях технических и естественных наук, так как позволяют строить наглядные и удобные для обработки модели сложных структур. При создании таких моделей граф наделяется некоторой семантикой, которая обычно выражается набором атрибутов, приписываемых его вершинам и дугам.

На практике преимущества графовых моделей во многих случаях становятся ощутимыми только при наличии хороших инструментальных средств их визуализации и обработки. Анализ существующих на сегодняшний день средств визуализации графов [1] показывает, что немаловажным параметром инструмента визуализации, определяющим круг его пользователей, является обеспечиваемое качество графического вывода. При этом можно выделить следующие основные критерии:

- качество получаемого изображения, т. е. точность соблюдения правил, заданных геометрическими атрибутами в представлении графа; иначе говоря, аккуратность построения рисунка;
- спектр средств (графических примитивов и их комбинаций, шрифтов, цветов и т.д.), используемых для создания изображения графа;
- гибкость задания графических параметров; например, система, в которой метку дуги можно разместить только у ее конца или начала, является менее гибкой, чем та, в которой эту метку можно разместить в любом месте вдоль линии дуги;
- набор возможностей пользователя по манипулированию изображением графа в процессе работы над ним; удобство этих манипуляций как следствие “интеллектуальности” системы.

Стоит отметить, что приведенные критерии часто противоречат друг другу, например: чем более широкий спектр средств используется для получения изображения и чем более аккуратное изображение мы пытаемся

[1] Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант № 00-07-90296) и Министерства образования РФ.

получить, тем больше машинного времени требуется на его построение, что может отрицательно сказаться на быстродействии системы и, соответственно, на возможностях пользователя быстро манипулировать изображением графа. Следовательно, нужно, с одной стороны, искать компромисс между удовлетворением данных критериев, с другой стороны, применять методики, позволяющие устранить противоречие между критериями. Основными параметрами системы, влияющими на достижение указанных целей, являются:

- схема задания геометрических атрибутов элементов графа;
- набор графических параметров элементов графа;
- методы построения графического представления по геометрическим атрибутам;
- оптимизация графического вывода;
- метод визуализации процесса редактирования графа.

При визуализации более сложных структур, таких как иерархические графы, появляются дополнительные проблемы, связанные с увеличением количества типов элементов графа и отношений между ними. Однако основные критерии качества остаются те же.

Иерархический граф состоит из вершин, дуг и фрагментов. Вершины и дуги представляют собой обычный граф, который может быть ориентированным или неориентированным. Каждый фрагмент ассоциируется с набором вершин, которые ему принадлежат. Фрагменты могут быть вложенными, если набор вершин одного является подмножеством набора вершин другого. Иначе пересекаться они не могут. Набор всех вершин определяет *главный* фрагмент иерархического графа. Более точное определение иерархических графов и других понятий, связанных с ними, можно найти в [2]. Различным вопросам визуализации иерархических графов посвящены работы [4–7].

Настоящая статья посвящена методам организации эффективного графического вывода в системах визуализации графов. Мы рассмотрим эти методы на примере системы Higres, являющейся визуализатором и редактором иерархических графовых моделей. В следующем разделе дается краткое описаие системы, ее основных возможностей и пользовательского интерфейса. Разд. 3 посвящен описанию схемы задания геометрических атрибутов элементов графа, использованной в системе. Указываются некоторые альтернативы и преимущества выбранной схемы. Построение изображения графа по данной схеме описано в разд. 4. В разд. 5 и 6 дается

описание методов оптимизации графического вывода и визуализации процесса редактирования графа, реализованных в системе.

2. СИСТЕМА HIGRES

Система Higres [8,9] предназначена для создания и визуализации иерархических графовых моделей, представляющих собой иерархические графы с заданной семантикой. Пример иерархического графа, созданного в системе Higres, приведен на рис. 1. Такие графы используются во внутреннем представлении IF-1 языка SISAL [3].

Семантика иерархического графа представляется в системе с помощью типов объектов. Каждый объект в графе принадлежит к какому-нибудь типу. Для каждого типа определяется набор меток. Каждая метка имеет тип данных, имя и несколько других параметров. Набор значений ассоциируется с каждым объектом графа в соответствии с набором меток, определенным для типа этого объекта. Вместе с разделением на типы эти значения задают семантику графа. Таким образом, пользователь может определять семантику, добавляя новые типы и их метки.

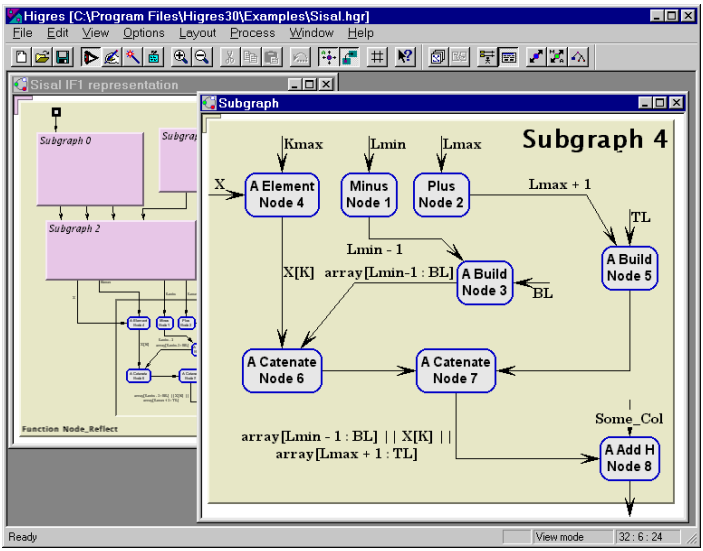

Рис. 1. Иерархический граф в системе Higres

В системе Higres каждый фрагмент визуально задается прямоугольником. Все вершины, принадлежащие фрагменту, располагаются внутри этого прямоугольника. Фрагменты так же, как и вершины, никогда не накладываются друг на друга. Каждый фрагмент может быть либо *закрытым*, либо *открытым*. В первом случае содержимое фрагмента скрыто от пользователя, во втором — видно внутри прямоугольника, представляющего данный фрагмент. Для каждого фрагмента можно открыть отдельное окно, в котором будет видно содержимое только этого фрагмента и его открытых подфрагментов.

Большая часть атрибутов объекта определяется его типом. Следовательно, семантически близкие объекты имеют сходное визуальное представление. Для визуализации меток объекта пользователь задает некоторый шаблон текста, в который вставляются значения меток каждого конкретного объекта.

Возможности визуализации, реализованные в системе, включают:

- различные формы и стили вершин;
- дуги в виде ломаных линий и гладких кривых;
- различные стили линий и стрелок дуг;
- выбор цвета для всех элементов графа;
- возможность задавать произвольный масштаб изображения;
- возможность перемещать текст дуги вдоль ее линии;
- позиционирование внешнего текста вершины в любом месте около ее границы;
- выбор шрифта для всех элементов графа;
- два файловых формата графического вывода;
- широкий набор опций визуализации.

Более полное описание пользовательского интерфейса системы и ее дополнительных возможностей можно найти в [8,9].

3. СХЕМА ЗАДАНИЯ ГЕОМЕТРИЧЕСКИХ АТРИБУТОВ И ВИЗУАЛЬНЫХ ПАРАМЕТРОВ ГРАФА

Все элементы графа в системе располагаются на некоторой абстрактной плоскости, части которой, соответствующие фрагментам графа, можно просматривать в окнах этих фрагментов.

3.1. Вершины. Основными параметрами вершины являются координаты ее центра, высота, ширина и форма. Следует отметить, что в некоторых случаях бывает удобно, чтобы центр вершины имел целые координаты, поэтому задавать вершины координатами углов окаймляющего прямоугольника не вседа правильно.

Каждая вершина может иметь две метки (вернее, два текста, каждый из которых может содержать несколько значений меток, но мы будем для простоты называть эти тексты метками): внутреннюю и внешнюю. Внутренняя метка располагается внутри вершины либо в ее центре, либо в верхнем правом углу, в зависимости от ее параметров. Внешняя метка располагается снаружи вблизи вершины следующим образом. Можно себе представить окаймляющий прямоугольник метки. Его размеры будут зависеть от длины текста и шрифта, следовательно, они будут разными для различных вершин. Метка располагается таким образом, чтобы данный прямоугольник касался каемки вершины. Направление вектора, соединяющего центры вершины и прямоугольника, задается отдельно для каждой вершины. При редактировании графа пользователь может изменять это направление, просто перемещая метку вокруг вершины. Для каждой формы вершины расположение внешней метки вычисляется по-разному, но фактически для задания этого расположения нам нужно знать только параметры вершины, текст метки, шрифт, которым она отображается, и направление соответствующего вектора. На рис. 2 приведены примеры позиционирования внешних меток для вершин различных форм.

Задание расположения внешней метки с помощью указанного вектора удобно потому, что при изменении размеров вершины и метки их приблизительное расположение относительно друг друга остается неизменным. В большинстве случаев при незначительных коррекциях метки или размеров вершины дополнительно подстраивать координаты метки не требуется. В системах, где метки позиционируются отдельно от вершин, это не так.

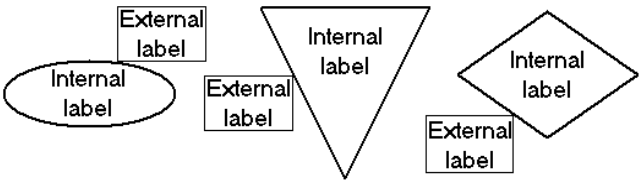

Рис.2. Позиционирование внешних меток вершин

3.2. Фрагменты. Фрагменты задаются координатами левого верхнего и правого нижнего углов. Предполагается, что все вершины и фрагменты, топологически лежащие внутри данного, имеют соответствующие координаты.

Как и вершины, фрагменты имеют две метки, однако отображаются несколько по-другому. Одна из меток отображается в том случае, когда фрагмент закрыт, другая — когда открыт. “Закрытая” метка фрагмента аналогична внутренней метке вершины и располагается либо в верхнем левом углу фрагмента, либо в его центре. “Открытая” метка тоже располагается внутри фрагмента, но ее можно позиционировать в любом месте около границы фрагмента. Как и в случае с внешней меткой вершины, позиционирование производится с помощью задания вектора, соединяющего центры прямоугольника фрагмента и окаймляющего прямоугольника метки.

3.3. Дуги. В системе Higres дуги изображаются как ломаными линиями, так и гладкими кривыми. Рассмотрим способ задания геометрического расположения дуги, имеющей форму ломаной линии. Нужно определить крайние точки линии, лежащие на границах изображений вершин, инцидентных данной дуге, и координаты всех точек сгиба дуги. Так как таких точек может быть произвольное количество, множество точек сгиба представляется списком. Граничные точки (т.е. точки входа в вершины) определяются одним из трех способов, задаваемым для каждой дуги отдельно:

1) ориентацией по центру вершины; при таком способе граничная точка выбирается так, чтобы крайнее звено дуги было направлено к центру вершины (см. фрагменты A и B на рис. 3);
2) ориентацией с помощью дополнительного вектора, задающего направление от центра вершины в точку входа дуги в вершину; вектор задается для двух концов дуги отдельно;
3) ортогональной ориентацией; данный способ используется для построения ортогональных изображений графов, т.е. таких, в которых все дуги представляются ломаными линиями, состоящими из вертикальных и горизонтальных звеньев; при редактировании графа строгую ортогонализацию внутренних звеньев легко произвести с помощью прямоугольной сетки, однако для крайних звеньев лучше всего подходит данный способ ориентации дуги (см. нижнее звено дуги в фрагменте C на рис. 3).

Важное свойство всех приведенных способов задания граничных точек состоит в том, что наиболее важные характеристики взаиморасположения вершины и входящей в нее дуги не изменяются при незначительных передвижениях объектов.

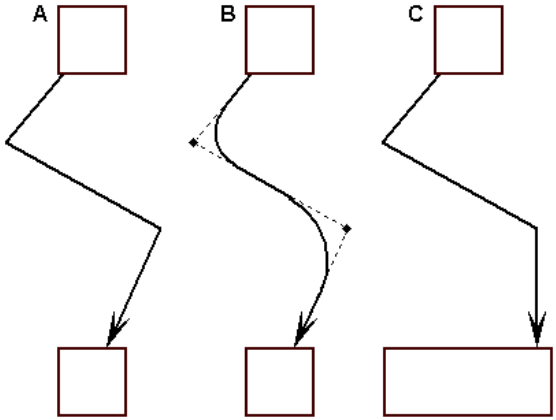

Рис.3. Задание расположения дуг с помощью точек сгиба.
Варианты прикрепления дуг к вершинам

Геометрическое расположение гладких дуг задается точно так же, как и для дуг, — в виде ломаных линий. Гладкая форма получается из ломаной путем “скругления” углов, т.е. замены дугами окружностей частей звеньев ломаной (см. фрагмент B на рис. 3).

Многие аналоги системы используют для изображения гладких дуг сплайны, составленные из кривых 2-го или 3-го порядков. Опыт показывает, что получаемые результаты в этом случае не имеют эстетических преимуществ над описанным выше способом. В то же время можно выделить два основных преимущества схемы, реализованной в системе Higres:

- более очевидный способ задания геометрического расположения (с помощью ориентирующей ломаной линии);
- более высокое быстродействие, связанное с использованием графических примитивов, отображаемых низкоуровневыми функциями операционной системы (в MS Windows нет функций для изображения произвольных кривых 2-го порядка).

Одной из наиболее сложных задач при проектировании схемы задания геометрических атрибутов графа является задание расположения меток дуг. Это связано с тем, что дуги геометрически могут представляться более разнообразно, чем вершины и фрагменты. В системе Higres эта задача решена следующим образом.

Так как расположение дуги задается координатами точек сгиба и точек крепления дуги к инцидентным вершинам, разумно именно к этим точкам привязывать координаты меток. Это обеспечит нам основное полезное свойство: при перемещении дуги метка перемещается вместе с ней, причем расположение ее будет сохраняться при незначительном изменении расположения дуги.

В то же время хочется достичь определенной гибкости в задании расположения метки, т. е. иметь возможность позиционировать ее лишь в месте одной из точек сгиба (или точки крепления) представляется не достаточно гибким способом. В системе Higres введено понятие точек привязки меток. Точками привязки метки для дуги являются:

а) точки сгиба;
б) точки входа дуги в инцидентные вершины;
в) точки, лежащие на серединах звеньев ломаной, представляющей дугу (для гладких дуг имеется в виду воображаемая ломаная линия, из которой получается линия дуги путем скругления углов).

Расположение метки дуги задается двумя параметрами: номером точки привязки и способом привязки. Существует 5 способов привязки:

а) точка привязки совпадает с серединой окаймляющего прямоугольника метки;
б) точка привязки совпадает с левым верхним углом окаймляющего прямоугольника метки;
в) точка привязки совпадает с левым нижним углом окаймляющего прямоугольника метки;
г) точка привязки совпадает с правым верхним углом окаймляющего прямоугольника метки;
д) точка привязки совпадает с правым нижним углом окаймляющего прямоугольника метки.

4. ПОСТРОЕНИЕ ИЗОБРАЖЕНИЯ ГРАФА

4.1. Вершины. На рис. 4 приведены различные варианты форм и стилей вершин, используемые в системе. Визуальные атрибуты, определяемые для каждого типа вершин, включают:

- форму (6 вариантов);
- стиль каемки (6 вариантов);
- цвет внутренней части (задается произвольно);
- цвет каемки (задается произвольно);
- шрифт, которым отображается внутренняя и внешняя метки.

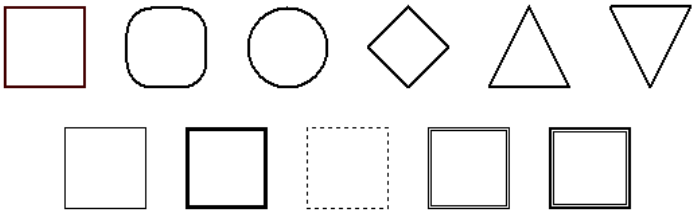

Рис. 4. Варианты форм и стилей вершин

4.2. Фрагменты. Изображение фрагмента имеет несколько меньше графических атрибутов, чем изображение вершины, а именно: для каждого типа фрагментов задается

- цвет фрагмента в закрытом состоянии;
- цвет фрагмента в открытом состоянии;
- шрифт, которым отображаются метки.

Закрытые и открытые фрагменты изображаются не только разными цветами, но и с каемками разного вида. Прямоугольники открытых фрагментов визуально представляются углублениями, а закрытых — выступами. Можно сказать, что закрытые фрагменты выглядят как участки плоскости, закрытые крышками (рис. 5).

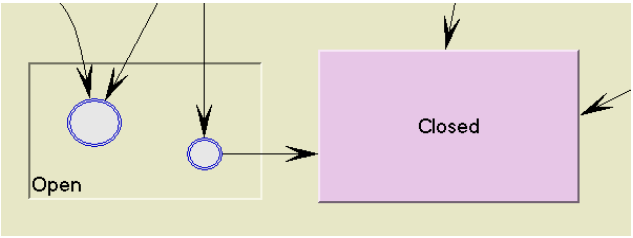

Рис .5. Изображение открытого и закрытого фрагментов

4.3. Дуги. Изображение дуг — наиболее сложная задача при выводе изображения графа. Функции, использующиеся исключительно для этой цели, в системе Higres в совокупности представляют собой около 500 строк кода на языке C++. Сложность изображения дуг обусловлена в первую очередь особенностями графического вывода дуг окружности в системе Windows. Во-первых, все параметры функции вывода дуги являются целочисленными. Следовательно, происходит двойное округление. Первый раз точные параметры дуги округляются при вызове функции, второй — исходя из уже округленных параметров производятся расчеты внутри этой функции, результаты которых опять округляются при выводе изображения. Во-вторых, вывод отрезков и дуг окружности существенно зависит от толщины линии. Обе проблемы фактически приводят к тому, что линии, из которых состоит дуга, плохо стыкуются друг с другом, что существенно ухудшает качество изображения. Чтобы избежать этого эффекта, необходимо при выводе корректировать координаты всех графических примитивов в зависимости от их относительного расположения и толщины линии.

Общая схема скругления углов ломаной линии при выводе гладких дуг приведена на рис. 6. Точки A, B и C — три последовательные точки сгиба дуги (A и C могут быть точками крепления дуги к вершине). Чтобы скруглить угол ABC, выбираем наибольший из отрезков AB и BC. Пусть это будет BC. Откладываем на нем точку N так, чтобы AB = BN. Координаты точки N легко вычисляются исходя из координат точек A, B и C. Делим отрезки AB и BN пополам. Получаем точки M и P. Их координаты также легко вычисляются. Наша цель — заменить дугой MP угол MBP. Для этого нужно вычислить центр окружности, вписанной в данный угол и касающейся его сторон в точках M и P. Этот центр будет лежать на пересечении перпендикуляров к AB и BC, проведенных через точки M и P соответственно. Уравнения прямых AB и BC легко получаются по координатам исходных

точек. Уравнения перпендикуляров к ним выводятся известным из аналитической геометрии способом.

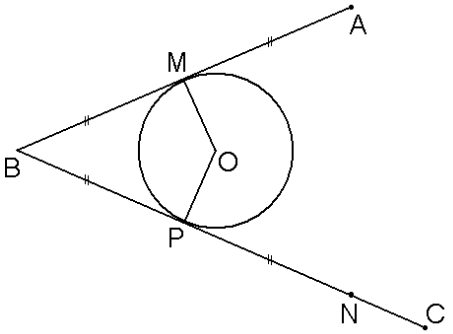

Рис. 6. Скругление угла при рисовании гладких дуг

Естественно, расчеты по данной схеме вычислений при реализации требуют отдельного рассмотрения всех экстремальных случаев (когда какие-либо из использующихся прямых параллельны осям координат).

Для ориентированных графов на концах дуг необходимо изображать стрелки, указывающие направление дуги. На рис. 7 приведены три варианта наконечников ориентированных дуг, используемых в системе. При рисовании наконечников производятся достаточно простые геометрические расчеты. Как и в случае с линией дуги, эти расчеты требуют корректирующей поправки для адаптации к функциям системы Windows и учета ширины линии.

Рис. 7. Варианты наконечников дуг

Заметим также, что для линий максимальной ширины (для дуг 3 пикселя) при рисовании крайнего звена основной линии дуги необходимо учитывать наличие наконечника, поскольку он имеет заостренную форму, сужающуюся до одного пикселя.

При изображении иерархических графов необходимо уметь изображать дуги, идущие внутрь закрытых фрагментов. Так как в данном случае вер-

шина, в которую идет дуга, скрыта внутри фрагмента, дуга должна заканчиваться на границе фрагмента. При этом если граф ориентированный, то дуга должна заканчиваться наконечником, указывающим на фрагмент (рис. 8). Естественно, изображение таких дуг должно быть как можно ближе к изображению тех же дуг при открытом состоянии фрагмента. В системе Higres используется следующий алгоритм для решения данной задачи.

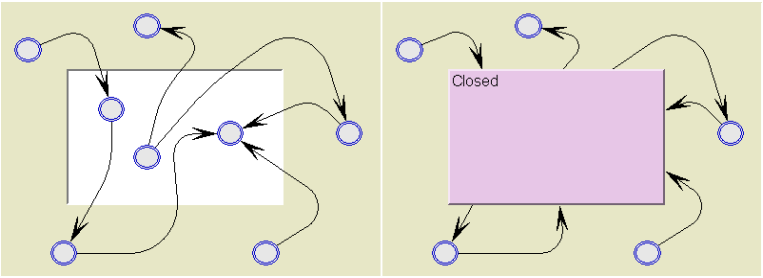

Рис. 8. Изображение дуг, идущих внутрь закрытых фрагментов

Если требуется изобразить дугу, одна из инцидентных вершин которой лежит внутри закрытого фрагмента, а другая за его пределами, перебираем отрезки, соединяющие последовательные точки сгиба данной дуги, начиная с конца, идущего от внешней вершины. Под точками сгиба будем понимать как непосредственные точки сгиба, так и точки крепления дуги к вершинам.

Как только при переборе дойдем до отрезка, пересекающего границу фрагмента, вычисляем координаты точки пересечения и заменяем последнюю из рассмотренных точек сгиба дуги на данную точку. Далее рисуем дугу по полученному набору точек сгиба.

Аналогичным способом решается задача изображения дуги, соединяющей вершины, принадлежащие двум различным закрытым фрагментам.

4.4. Метки. При изображении внешних меток вершин и меток дуг возникает проблема цвета фона. Для каждого типа вершин и дуг используется параметр, который указывает, заполнять ли фон при отображении меток данного типа объектов или нет. Отображение фона необходимо использовать, например, в случае, если текст метки накладывается непосредственно на линию дуги. Тогда текст может “слиться” с этой линией, если выводить его с выключенной опцией фона. С другой стороны, если эта опция включена, то возникает следующая проблема. Метка может накладываться на произ-

вольный участок плоскости. Теоретически он может содержать произвольные объекты произвольного цвета. Следовательно выбор цвета фона неоднозначен. Для наиболее правильного выбора этого цвета используется следующий метод. Просчитывается окаймляющий прямоугольник метки и берется 5 образцов цвета фона из точек, лежащих под углами данного прямоугольника и под его центром. Цвет, присутствующий в большинстве перечисленных точек, используется в качестве цвета фона. В большинстве случаев данный подход дает оптимальный результат.

4.5. Масштабирование изображения. Важным параметром системы визуализации является ее способность создавать качественные изображения графа при любом масштабе. Основная сложность при масштабировании состоит в том, что практически невозможно и в большинстве случаев нерационально пытаться масштабировать все элементы изображения одинаковым образом. В системе Higres масштабирование производится по следующей схеме.

- Размеры вершин и фрагментов изменяются при изменении масштаба.
- Толщина линий, включая линии дуг, каемок вершин и фрагментов, не изменяется.
- Шрифты масштабируются средствами Windows, что дает не полностью корректный, но приемлемый результат.
- Размеры наконечников дуг либо изменяются пропорционально масштабу, либо остаются неизменными в зависимости от соответствующего параметра типа дуги.

Для получения более качественного изображения координаты всех объектов при масштабировании переводятся в целые числа только при получении итогового расположения элементов графа.

5. ОПТИМИЗАЦИЯ ГРАФИЧЕСКОГО ВЫВОДА

Правильно выбранная схема задания геометрических характеристик элементов графа и применение адекватных методов построения изображения по этой схеме создают основу для получения быстродействующей системы визуализации. Однако необходимо также уделить внимание последнему этапу получения изображения — непосредственно графическому вы-

воду. Так как задача сводится к отображению на экране (или другом подобном устройстве вывода) уже просчитанного с точностью до пикселя изображения, решение существенно зависит от особенностей операционной системы.

Метод графического вывода, реализованный в системах MS Windows, можно кратко описать следующим образом. В каждом окне есть клиентская часть, за перерисовку которой отвечает приложение, создавшее данное окно, т. е. приложение предоставляет операционной системе функцию, которую нужно вызывать каждый раз, когда требуется перерисовать клиентскую область окна или какую-либо ее часть. Это происходит либо в случае, когда другое окно перемещается поверх данного и требуется перерисовать открывшийся участок, либо по инициативе самого приложения в случае, когда нужно обновить содержимое окна.

В обоих случаях перерисовываемый участок имеет прямоугольную форму. Таким образом, задача, решаемая функцией перерисовки, сводится к выводу всех элементов изображения, находящихся внутри некоторого прямоугольника с заданными координатами.

Самое элементарное решение данной задачи (к сожалению, часто используемое на практике) состоит в том, чтобы перерисовать все изображение. Механизм перерисовки Windows устроен так, что при этом фактически на экране будет отображаться только нужная часть. Оптимизированный вариант функции должен попытаться не тратить время на рисование элементов, лежащих за пределами области перерисовки. Именно такой метод применен в системе Higres.

Другая важная оптимизация состоит в следующем. Далеко не все действия пользователя, требующие перерисовки изображения графа, требуют также изменения графического представления элементов графа. Например, если пользователь просто прокручивает изображение в окне, в принципе нет необходимости пересчитывать расположение графических примитивов, из которых состоят изображения элементов графа. Этот пересчет нужен только при изменении графа (включая перемещение элементов и модификацию любых графических параметров) и масштаба изображения.

В системе Higres реализована оптимизированная трехэтапная схема графического вывода. Для построения изображения фрагмента графа (напомним, что нам всегда требуется нарисовать какой-либо фрагмент, так как именно фрагменты ассоциированы с окнами) на первом этапе строится множество графических примитивов (с координатами и прочими атрибутами), из которых состоит изображение. При этом рассматриваются только

вершины и фрагменты графа, лежащие внутри данного, и только дуги, инцидентные этим вершинам. Это множество запоминается в виде списка и пересчитывается только тогда, когда происходит изменение графа, способное повлиять на изображение данного фрагмента. Заметим, что такое изменение может быть произведено пользователем и внутри окна другого фрагмента. Для определения области влияния модификаций графа применяется специальный алгоритм, который позволяет избежать излишних пересчетов и перерисовок.

На втором этапе строится изображение перерисовываемого прямоугольника в памяти компьютера. Наконец, на третьем этапе сформированное в памяти изображение переносится в нужное место экрана. Необходимость промежуточного этапа рисования в памяти обусловлена двумя причинами. Во-первых, операционная система Windows устроена таким образом, что процесс рисования в памяти происходит быстрее, чем непосредственно на экране, даже если учесть время, потраченное на дополнительную операцию копирования на экран. Во-вторых, при таком подходе обновленное изображение появляется на экране мгновенно, даже если система потратила некоторое время на его построение.

Применение приведенной схемы оптимизации графического вывода позволяет прежде всего сделать максимально комфортной для пользователя процедуру просмотра графа, поскольку традиционная задержка при перерисовке прокручиваемого изображения графа сведена к минимуму. Хотя время этой задержки в любом случае линейно зависит от количества элементов в фрагменте, линейная составляющая начинает сказываться только на достаточно больших фрагментах (порядка нескольких сотен вершин).

6. ВИЗУАЛИЗАЦИЯ ПРОЦЕССА РЕДАКТИРОВАНИЯ ГРАФА

Система Higres является не только визуализатором графовых моделей, но и полноценным графовым редактором. Это означает, что с помощью нее можно вносить в модель произвольные изменения. Многие модификации графа должны производиться визуально. К таким модификациям относятся следующие:

- добавление и удаление вершин, фрагментов и дуг;
- перемещение вершин, фрагментов и дуг;
- позиционирование меток элементов графа.

Практически во всех системах редактирование осуществляется с помощью выделения одного или нескольких объектов и последующих манипуляций с выделенными объектами с помощью мыши, например, можно выделить вершину и далее переместить ее в другое место изображения. При реализации этого подхода возникают две основные проблемы.

1. Выделение должно производиться таким образом чтобы пользователь мог легко отличать выделенные объекты от невыделенных. Это естественное требование достаточно легко соблюсти, когда система позволяет выделять только один или несколько однотипных объектов. В нашем случае имеется несколько видов объектов (вершины, фрагменты и дуги), которые, кроме того, могут накладываться друг на друга. Для того чтобы можно было выделять несколько объектов разных видов одновременно, необходимо использовать различные способы выделения для различных видов объектов, а также производить выделение каждого объекта таким образом, чтобы оно как можно точнее указывало на его расположение.

2. В процессе изменения пользователем геометрических параметров редактируемого элемента крайне нежелательно производить полную перерисовку изображения графа, поскольку даже при всех возможных оптимизациях это очень дорогостоящий по времени процесс. В то же время пользователь должен иметь возможность отслеживать визуальные параметры объектов во время редактирования. Например: если пользователь перемещает группу вершин с помощью мыши, то, с одной стороны, он должен видеть их текущую позицию в каждый момент времени, с другой стороны, если для этого при каждом сдвиге перерисовывается все изображение фрагмента, интерфейс системы будет явно слишком медленным при достаточно большом размере графа.

На рис. 9 показаны различные варианты выделения объектов при редактировании в системе Higres: выделение нескольких объектов различных видов (A), выделение одного фрагмента (B), вершины (C) и дуги (D). Во всех случаях выделение производится пунктирной линией. В первом случае пунктиром показываются только окаймляющие прямоугольники фрагментов и вершин, а также линии дуг. В остальных случаях, когда выделен только один объект, выделение имеет дополнительные свойства. Для фрагмента прямоугольником выделяется его метка (пользователь может позиционировать ее с помощью мыши), а также в центре прямоугольника фрагмента изображается перекрестие для того, чтобы сделать выделение фрагмента отличным от выделения вершины. Для вершины дополнительно выделяется

текст ее внешней метки. Как и в случае с фрагментами, этот текст можно позиционировать мышью. Для дуги дополнительно выделяется текст метки и все ее точки сгиба.

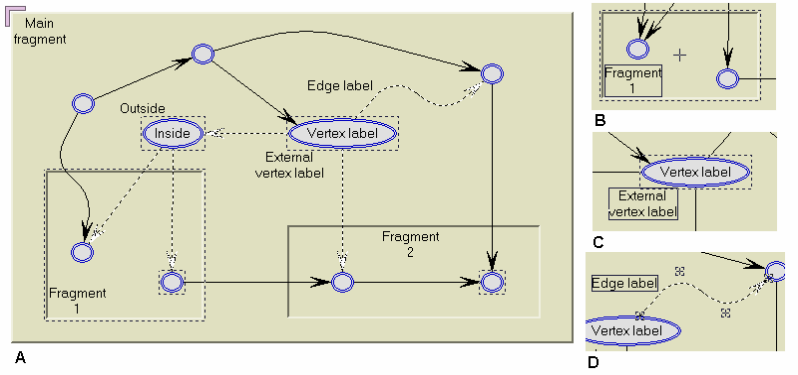

Рис. 9. Выделение элементов графа при редактировании

Вторая из приведенных выше проблем решается в системе Higres следующим образом. Изображение выделенного объекта получается из изображения невыделенного добавлением дополнительных пунктирных линий, которые рисуются методом инвертирования изображения, т. е. если дважды нарисовать одну и ту же линию, получим исходное изображение. Таким образом, чтобы убрать с изображения выделение, достаточно просто еще раз его отобразить. Если нам нужно переместить выделение (например, если пользователь перемещает выделенные объекты с помощью мыши), достаточно вывести его один раз со старыми координатами (что вернет нам изображение без выделения), а затем вывести его с новыми координатами.

Так как выделение состоит из набора графических примитивов, разумно применить к нему ту же оптимизацию, что и к основному изображению, а именно: запоминать отдельно множество примитивов, используемых для вывода выделения, и пересчитывать его только тогда, когда оно существенно изменяется. В отличие от основного изображения выделение выводится непосредственно на экран, поскольку в данном случае промежуточный этап вывода в память не может служить оптимизацией. Выделение, как правило, состоит из достаточно маленького числа линий, поэтому их вывод занимает намного меньше времени, чем копирование всего изображения из памяти на экран.

7. ЗАКЛЮЧЕНИЕ

Мы рассмотрели методы организации графического вывода в системах визуализации графовых объектов. Отмечены и классифицированы основные проблемы, возникающие при проектировании таких систем, даны общие методы их решения и описан конкретный вариант реализации данных подходов в системе Higres, являющейся визуализатором и редактором иерархических графовых моделей.

ЛИТЕРАТУРА

1. **Лисицын И.А.** Системы визуализации и редактирования графовых объектов. — Новосибирск: ИСИ СО РАН. — (В печати).
2. **Касьянов В.Н.** Иерархические графы и графовые модели: вопросы визуальной обработки // Проблемы систем информатики и программирования. — Новосибирск, 1999. — С. 7–32.
3. **Густокашина Ю.В., Евстигнеев В.А.** IF1 — промежуточное представление SISAL-программ // Проблемы конструирования эффективных и надежных программ. — Новосибирск, 1995. — С. 70–78.
4. **Eades P., Feng Q.W.** Drawing clustered graphs on an orthogonal grid // Proc. of Graph Drawing 97. — Berlin a.o.: Springer Verlag, 1997. — P. 182–193. — (Lect. Notes Comput. Sci.; Vol. 1353).
5. **Eades P., Feng Q.W.** Multilevel visualization of clustered graphs // Proc. of Graph Drawing 96. — Berlin a.o.: Springer Verlag, 1996. — P. 101–112. — (Lect. Notes Comput. Sci.; Vol. 1190).
6. **Eades P., Feng Q.W., Lin X**. Straight-line drawing algorithms for hierarchical graphs and clustered graphs // Proc. of Graph Drawing 96. — Berlin a.o.: Springer Verlag, 1996. — P. 113–128. — (Lect. Notes Comput. Sci.; Vol. 1190).
7. **Feng Q.W., Cohen R., Eades P**. How to draw a planar clustered graph // In COCOON '95. — Berlin a.o.: Springer Verlag, 1995. — P. 21–31. — (Lect. Notes Comput. Sci.; Vol. 959).
8. **Lisitsyn I.A. Kasyanov V.N**. Higres — Visualization system for clustered graphs and graph algorithms // Proc. of Graph Drawing 99. — Berlin a.o.: Springer Verlag, 1999. — P. 82–89. — (Lect. Notes Comput. Sci.; Vol. 1731).
9. **Лисицын И.А.** Применение системы HIGRES для визуальной обработки иерархических графовых моделей // Проблемы систем информатики и программирования. — Новосибирск, 1999. — С.64–77.