Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

Укладка графа на плоскости

АВТОРЕФЕРАТ

дипломной работы

студентки 6 курса 631 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Гадратовой Полины Михайловны

Научный руководитель

доцент, к.ф.-м.н. _____ А.В. Жаркова

31.12.2016 г.

Заведующий кафедрой

профессор, к.ф.-м.н. _____ В.Н. Салий

31.12.2016 г.

Саратов 2017

ВВЕДЕНИЕ

Теория графов является обширным и интересным разделом дискретной математики. Графы используются во многих науках: химии, электротехнике, социологии, теории сетей.

В теории графов существует множество актуальных проблем. Одной из таких проблем является проблема построения плоского изображения планарного графа. Планарные графы широко используются в схемотехнике при создании печатных плат, при построении сетей, проектировании дорог.

Целью данной дипломной работы является изучение алгоритмов укладки графа на плоскости и создание программного продукта для построения плоского изображения планарных графов.

Для достижения поставленной цели требуется решить следующие задачи:

- изучить теоретические сведения, касающиеся планарных графов;
- рассмотреть алгоритмы проверки графа на планарность;
- рассмотреть алгоритмы, позволяющие построить плоское изображение планарного графа.

Дипломная работа состоит из введения, 4 разделов, заключения, списка использованных источников и 1 приложения. Общий объем работы – 98 страниц, из них 43 страницы – основное содержание, включая 35 рисунков, список использованных источников из 16 наименований.

КРАТКОЕ СОДЕРЖАНИЕ

Во введении ставится цель работы: изучение алгоритмов укладки графа на плоскости и создание программного продукта для построения плоского изображения планарных графов. Для достижения поставленной цели требуется решить следующие задачи: изучить теоретические сведения, касающиеся планарных графов, рассмотреть алгоритмы проверки графа на планарность и рассмотреть алгоритмы, позволяющие построить плоское изображение планарного графа.

В разделе 1 «Необходимые определения» приводятся некоторые необходимые определения, используемые в работе, в том числе орграф, граф, подграф, блок графа, цикл в графе и т.д. [1, 2, 3, 4]

В разделе 2 «Построение плоской укладки графа» приводятся необходимые определения, касающиеся планарных графов, алгоритмы построения плоского изображения дерева, графа, проверки графа на планарность.

В подразделе 2.1 «Планарные графы» рассматриваются основные определения и теоремы, связанные с планарными графами, такие как формула Эйлера со следствиями, теоремы о планарности графа. [1, 4, 5, 6, 7]

В подразделе 2.2 «Построение плоского изображения дерева» рассматриваются некоторые необходимые определения, касающиеся деревьев, а также алгоритм, позволяющий построить плоское изображение дерева.

Произвольно выберем некоторую вершину v_0 (корень дерева). Если k – расстояние от v_0 до наиболее удаленных от корня вершин, проведем $k+1$ горизонталей, нумеруя их снизу вверх числами $0, 1, \dots, k$. На нулевой горизонтали поместим вершину v_0, а на горизонтали 1 расположим произвольным образом вершины, смежные с v_0 (т.е. находящиеся на расстоянии 1 от корня). Пусть это будут (слева направо) вершины $v_{i_1}, v_{i_2}, \dots, v_{i_s}$. На горизонтали 2 разместим вершины, удаленные на расстояние 2 от корня v_0, таким образом, что сначала будут идти вершины, смежные с v_{i_1}, за ними –

вершины, смежные с v_{i_2} и т.д. Завершив, в соответствии с этим правилом, размещение всех вершин на горизонталях, соединим каждую из них прямолинейными отрезками со смежными с ней вершинами следующего уровня (если таковые есть). [1]

В подразделе 2.3 «Алгоритмы построения плоского изображения графа» рассматриваются некоторые алгоритмы, позволяющие строить плоское изображение планарного графа.

Алгоритмы на основе физической модели в своей основе имеют соответствие некоторой физической модели. [9] К алгоритмам на основе физической модели можно отнести такие алгоритмы, как «пружинный» метод, «силовой» метод, метод отжига и другие.

В «пружинном методе» графу сопоставляется некоторая упругая механическая система, в которой точкам соответствуют вершины графа, а «пружинам» – упругим силовым связям – ребра графа; условие равновесия системы соответствует минимуму штрафной функции. Данный алгоритм имеет полиномиальную сложность. [8] В «силовом» методе граф рассматривается как система пружин - ребер и шарниров с вершинами – одноименными электрическими зарядами, между которыми действуют силы отталкивания. [9]

Метод отжига – это техника оптимизации, использующая упорядоченный случайный поиск на основе аналогии с процессом образования веществом кристаллической структуры с минимальной энергией при охлаждении. [10]

Аналитические алгоритмы представляют собой последовательность преобразований графа, приводящих к его укладке. Данные алгоритмы имеют гарантированный результат, но плохо поддаются модификации. Примерами аналитических алгоритмов являются такие алгоритмы как GIOTTO и гамма-алгоритм.

GIOTTO является одним из наиболее эффективных аналитических алгоритмов. Он строит *ортогональное* плоское изображение планарного графа, то есть такое изображение, в котором ребра графа изображаются как ломаные

линии – совокупности горизонтальных и вертикальных отрезков. Алгоритм GIOTTO практически не поддается модификации. [6]

Гамма-алгоритм представляет собой процесс последовательного присоединения к некоторому уложенному подграфу $\tilde{G}$ графа G новой цепи, оба конца которой принадлежат $\tilde{G}$, притом цепь разбивает одну из граней графа $\tilde{G}$ на две; в качестве начального плоского графа $\tilde{G}$ выбирается любой простой цикл графа G. Процесс продолжается до тех пор, пока не будет построен плоский граф, изоморфный графу G, или пока некоторую цепь окажется невозможно присоединить к подграфу $\tilde{G}$ – в этом случае граф G не является планарным. [12] На вход подаются графы, обладающие следующими свойствами: граф связный, граф имеет хотя бы один цикл, граф не имеет мостов. [6] Доказательства корректности гамма-алгоритма приводятся в источниках [6, 12].

Генетический алгоритм укладки графа, предложенный в [7], имеет блочную структуру, каждый блок является отдельным фрагментом, входные данные для которого предоставляет предыдущий блок (за исключением начального блока). Первым блоком является блок ввода исходных данных. Следующий блок – блок генерации циклов, который производит генерацию циклов заданной длины, выполняя это действие столько раз, сколько потребует от него блок анализа. Генерация циклов осуществляется на основе поиска в глубину. Полученные данные являются исходными для начала процедуры анализа. Результатом ее работы является вывод о планарности или непланарности графа. Основным блоком алгоритма является блок генетических операторов, его работа подробно описана в [7, 12]. Последним блоком алгоритма является блок укладки. [7] При фиксированных значениях размера популяции и числа генераций (поколений) алгоритм имеет полиномиальную теоретическую пространственную и временную сложность, а значит имеет практическую полезность. [12]

В подразделе 2.4 «Проверка графа на планарность» рассматривается алгоритм, с помощью которого можно проверить планарность графа. Задача данного алгоритма состоит в том, чтобы найти множество циклов графа с n

вершинами и m ребрами и выделить такие $m - n + 2$ циклов, что каждое ребро графа будет принадлежать точно двум из них, причем графы, удовлетворяющие условию $m \leq n + 2$ являются заведомо планарными графами, а графы, удовлетворяющие условию $m > 3(n - 2)$ являются заведомо непланарными графами. [7]

В генетическом алгоритме [7, 12] имеется блок анализа промежуточных данных (блок анализа), с помощью которого, в том числе, происходит проверка графа на планарность. Алгоритм этой проверки можно привести в следующем виде:

1) провести поиск всех циклов графа длины k, где $k \geq 3$, то есть таких циклов, что никакие два из них не содержат один и тот же набор ребер;

2) подсчитать число сформированных циклов $N_{C_k} = |C_k|$ и число повторений N_{m_j} для каждого ребра m_j, где $1 \leq j \leq M$;

3) если $N_{C_k} \geq m - n + 2$ и $(\forall j \in M)(N_{m_j} \geq 2)$, то переходим к шагу 3, иначе возвращаемся к генерации циклов и генерируем циклы длины $k + 1$;

4) для первых $m - n + 2$ циклов из множества C_k подсчитать общее число ребер N_m, принадлежащих этим циклам:

$$N_m = \sum_{k=3}^{K}(k|C_k|),$$

где k – длина цикла, $|C_k|$ – мощность множества циклов длины k, K – длина самого длинного из сгенерированных циклов, вычисляемая по формуле $K = k + s$, где s – число генераций циклов;

5) значение N_m сравнить с удвоенным числом ребер графа G. Если $N_m > 2m$, то граф непланарный;

6) если $N_m \leq 2m$, то к множеству добавить циклы длины $k + 1$. Если циклы длины $k + 1$ отсутствуют, то вернуться к генерации циклов длины $k + 2$, в противном случае перейти к шагу 7;

7) конец работы алгоритма.

В программной реализации будем использовать модификацию данного алгоритма. Для выполнения шагов 2–7 удобно использовать матрицу циклов $B =$

$\|b_{ij}\|_{c\times m}$, где c – число циклов, а m – число ребер графа, причем $b_{ij} = 1$, если ребро m_j принадлежит циклу c_i, и $b_{ij} = 0$ – в противном случае. [7] Модифицированный алгоритм будет отличаться от приведенного тем, что

1) будем искать все неповторяющиеся циклы графа единовременно, что обусловлено простотой реализации. Не будем говорить о генерации циклов на шагах 3, 6;

2) не будем считать число сформированных циклов N_{c_k} на шаге 2 и не будем проверять количество циклов на соответствие условию $N_{c_k} \geq m - n + 2$ на шаге 3. Данная проверка основана на теореме 1; очевидно, количество всех неповторяющихся циклов графа больше, чем количество его граней;

3) на шаге 4 сразу будем искать такой набор из $m - n + 2$ циклов, что общее число ребер в нем будет равно удвоенному числу ребер графа;

4) если на шаге 6 $N_m \leq 2m$, будем возвращаться к шагу 4 для поиска следующего набора циклов;

5) введем новый шаг, который будем считать шагом 7: для найденного набора из $m - n + 2$ циклов, для которого $N_m = 2m$, будем проверять, что каждое ребро графа входит точно в два из этих циклов. Если это условие выполнено, граф является планарным, если условие не выполняется, возвращаемся к шагу 4 и ищем следующие $m - n + 2$ циклов. В случае, если все возможные наборы из $m - n + 2$ циклов просмотрены, и нет таких наборов, которые соответствуют условию на шаге 7 модификации, граф не является планарным. Введенная на шаге 7 проверка основана непосредственно на теореме 3.

В разделе 3 «Некоторые программные средства для проверки планарности и построения плоских изображений планарных графов» рассматриваются некоторые из существующих систем, позволяющих работать с графами, а именно система Draw [13], пакет Graphviz [14] и «Графоанализатор» [15].

В разделе 4 «Программная реализация» рассматривается разработанная и реализованная в рамках данной дипломной работы программа «BuildPlanar»,

выполняющая построение плоского изображения дерева (см. подраздел 2.2) и проверку графа на планарность (см. подраздел 2.4). При написании этой программы использовалась среда разработки Borland C++ Builder 6, благодаря чему «BuildPlanar» не имеет высоких системных требований и может быть запущена на любом компьютере, поддерживающем ОС Windows 7 или новее. Для корректной работы на компьютере программе может требоваться midas.dll (Borland MIDAS Component Package). Данный файл прилагается к программе, при необходимости его нужно лишь зарегистрировать в Windows с помощью системной утилиты regsvr32.

При запуске программы «BuildPlanar» открывается окно, из которого можно создать новый граф или открыть сохраненный ранее граф, нажав соответствующую кнопку.

В редакторе есть три вкладки. Выбрать нужную можно из выпадающего списка наверху. Ввод графа в программу следует производить во вкладке BASE_INPUT, результаты последнего построения можно увидеть на вкладке BASE_OUTPUT. Вкладка BASE_INTEREDIATE содержит служебную информацию программы.

На вкладке BASE_INPUT окна редактора можно добавлять и удалять вершины, ребра графа, перемещать вершины. Добавить в граф вершину можно путем нажатия Ins на клавиатуре или из контекстного меню, вызвав его правой кнопкой мыши. Удалить вершину можно путем выделения нужной вершины и двойного нажатия на клавиатуре Del. Если к удаляемой вершине были проведены ребра, они также удаляются. Между двумя любыми различными вершинами графа может быть создано ребро, для чего нужно выделить щелчком левой кнопки мыши нужную вершину, а затем щелкнуть по второй вершине правой кнопкой мыши. Удалить существующее ребро можно путем его выделения и нажатия на клавиатуре Del.

В окне редактора щелчком правой кнопки мыши по свободной области можно вызвать контекстное меню. Данное меню содержит несколько пунктов. Пункт «Создать вершину» позволяет создать вершину графа, пункт «Удалить

вершину /ребро» позволяет удалить выделенные ранее вершину или ребро графа. Пункт «Проверить планарность графа» позволяет запустить проверку планарности в графе. Пункт «Построить плоское изображение дерева» позволяет построить плоское изображение дерева, заданного пользователем. Пункт «Обновить» позволяет обновить изображение в окне редактора. Пункт «Добавить комментарий» позволяет присвоить графу в окне сохранения некоторый комментарий.

Как было сказано выше, пункт «Проверить граф на планарность» позволяет проверить планарность графа. Стоит заметить, что на больших графах проверка на планарность работает дольше. После нажатия пункта «Проверить граф на планарность» (или кнопки L на клавиатуре) программа выводит окно, сообщающее о планарности или непланарности графа или о невозможности проверить граф на планарность.

Пункт меню «Построить плоское изображение дерева» (или нажатие клавиши T) позволяет построить плоское изображение дерева. Перед тем, как выбрать данный пункт, следует выбрать корень дерева, которое необходимо построить. Построенное изображение будет приведено на вкладке BASE_OUTPUT. В том случае, если граф не является деревом, то есть, если в графе присутствуют циклы, программа выдаст сообщение о том, что граф не является деревом. Если при построении не выбран корень дерева, программа выдаст соответствующее сообщение.

В программе есть краткая справка о работе с программой.

В заключении содержатся основные результаты работы и краткие выводы по ним.

В списке использованных источников приводятся 16 наименований.

В приложении А «Листинг программы» приведен код созданного программного продукта на 55 листах.

ЗАКЛЮЧЕНИЕ

В теории графов задача построения плоского изображения планарного графа является интересной и важной задачей, которая актуальна во многих областях деятельности.

В ходе данной дипломной работы были изучены наиболее важные понятия теории графов, касающиеся планарности графов. Были рассмотрены алгоритм построения плоского изображения дерева, алгоритмы построения плоского изображения планарного графа, такие как «метод отжига», GIOTTO, гамма-алгоритм, алгоритм проверки графа на планарность. Также были рассмотрены программы, позволяющие строить плоское изображение планарного графа или проверять граф на планарность.

В ходе проделанной работы была разработана и реализована программа, позволяющая строить плоское изображение дерева, а также проверять граф на планарность. Реализованная в данной программе проверка графа на планарность отличается от рассмотренных аналогов.

Таким образом, все поставленные задачи были решены.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Богомолов, А. М. Алгебраические основы теории дискретных систем / А. М. Богомолов, В. Н. Салий. М. : Наука. Физматлит, 1997. 368 с.

2 Абросимов, М. Б. О реконструируемости малых турниров [Электронный ресурс] / М. Б. Абросимов, А. А. Долгов // Изв. Сарат. ун-та. Нов. сер. Сер. Математика. Механика. Информатика. 2009. Т. 9, вып. 2. С. 94–98. Загл. с экрана. Яз. рус.

3 Kant, G. Two algorithms for finding rectangular duals of planar graphs [Электронный ресурс] / G. Kant, X. He // 19th International Workshop, WG '93 Utrecht, The Netherlands, June 16–18, 1993 Proceedings. Springer Berlin Heidelberg, 1994. P. 396–410. Загл. с экрана. Яз. англ.

4 Харари, Ф. Теория графов [Электронный ресурс] / Ф. Харари ; пер. В. П. Козырева. М. : УРСС, 2003. 296 с. Загл. с экрана. Яз. рус.

5 Окулов, С. М. Дискретная математика. Теория и практика решения задач по информатике : учебное пособие [Электронный ресурс] / С. М. Окулов. М. : БИНОМ. Лаборатория знаний, 2012. 424 с. Загл. с экрана. Яз. рус.

6 Иринёв, А. Алгоритм плоской укладки графов [Электронный ресурс] / А. Иринёв, В. Каширин // Дискретная математика: алгоритмы [Электронный ресурс] : [сайт]. URL: http://rain.ifmo.ru/cat/data/theory/graph-coloring-layout/layout-2006/article.pdf (дата обращения: 02.11.2016). Загл. с экрана. Яз. рус.

7 Биоинспирированные методы в оптимизации [Электронный ресурс] / Л. А. Гладков [и др.]. М. : ФИЗМАТЛИТ, 2009. 384 с. Загл. с экрана. Яз. рус.

8 Соколов, Г. В. Анализ алгоритмов автоматической укладки графов на плоскости в рамках задачи визуализации моделей на графах [Электронный ресурс] / Г. В. Соколов // Вестник Пермского национального исследовательского политехнического университета. Прикладная математика и механика. 2010. № 15. С. 162–171. Загл. с экрана. Яз. рус.

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

10 Лопатин, А. С. Метод отжига [Электронный ресурс] / А. С. Лопатин // Математико-механический факультет СПбГУ [Электронный ресурс] : [сайт]. URL: http://www.math.spbu.ru/user/gran/sb1/lopatin.pdf (дата обращения: 02.11.2016). Загл. с экрана. Яз. рус.

11 Коротков, М. А. Разработка и реализация алгоритма укладки диаграмм состояний : бакалаврская работа [Электронный ресурс] / М. А. Коротков // Сайт по автоматному программированию и мотивации к творчеству [Электронный ресурс]. URL: http://is.ifmo.ru/papers/mkorotkov-bachelor.pdf (дата обращения: 21.12.2016). Загл. с экрана. Яз. рус.

12 Лекции по теории графов / В. А. Емеличев [и др.]. М. : Наука. Гл. ред. физ.-мат. лит., 1990. 384 с.

13 Гладков, Л. А. Генетический алгоритм планаризации на основе матрицы цепей [Электронный ресурс] / Л. А. Гладков // Известия ТРТУ. Тематический выпуск: материалы международной конференции «Интеллектуальные САПР», 1999. Т. 13, № 3. С. 133–139. Загл. с экрана. Яз. рус.

14 Welcome to uDraw(Graph) [Электронный ресурс] // Universität Bremen [Электронный ресурс] : [сайт]. URL: http://www.informatik.uni-bremen.de/uDrawGraph/en/home.html (дата обращения: 01.11.2016). Загл. с экрана. Яз. англ.

15 Graphviz | Graphviz - Graph Visualization Software [Электронный ресурс] // Graphviz [Электронный ресурс] : [сайт]. URL: http://www.graphviz.org/ (дата обращения: 01.11.2016). Загл. с экрана. Яз. англ.

16 Справка по программе Графоанализатор 1.3 [Электронный ресурс] // Графоанализатор – среда для работы с графами [Электронный ресурс] : [сайт]. URL: http://grafoanalizator.unick-soft.ru/help/html/helpgraf1.3.html (дата обращения: 01.11.2016). Загл. с экрана. Яз. рус.