Информационные процессы, Том 19, № 2, 2019, стр. 159–169

Моделирование графов с различными видами достижимости с помощью языка Python [1]

В.М. Антонова[*,**], **Б.М. Захир**[*], **Н.А. Кузнецов**[**]

[*] *Московский государственный технический университет им. Н.Э. Баумана, Москва, Россия*
[**] *Институт радиотехники и электроники им. В.А. Котельникова РАН, Москва, Россия*

Поступила в редколлегию 15.05.2019

Аннотация—Целью данной статьи является исследование различных вариантов ресурсных сетей с различными типами ограничений на достижимость.

КЛЮЧЕВЫЕ СЛОВА: графы, потоки на графах, магнитная достижимость, Барьерная достижимость

1. ВВЕДЕНИЕ

Задачи на графах являются самыми актуальными задачами современных телекоммуникационных сетей. Однако современные телекоммуникационные сети совмещают в себе различные технологии передачи данных, которые в свою очередь имеют различные характеристики передачи.

В большинстве прикладных задач изучается не сам граф, а процессы на нём, как правило маршрутизация, которая непосредственно связана с достижимостью некоторых вершин и ребер. В работах [1–3] были определены и изучены различные типы нестандартной достижимости, в частности: смешанная, магнитная, вентильная, барьерная и т.д. Все эти типы ограничений формулировались в виде некоторого набора требований, предъявляемых к дугам, допустимых этим типом достижимости путей. В прикладных задачах часто встречаются конфигурации с ограничением не на дуги графа, а на его вершины. Задачи с ограничением на дуги будут рассматриваться в рамках данной статьи.

Отметим, что такая задача не всегда может иметь решение. Например, если в ориентированном графе есть петля, метка которой – отрицательное число, то по этой петле можно проходить сколько угодно раз и тем самым уменьшать сумму меток дуг пути, включающего эту петлю, до любого наперед заданного значения.

В телекоммуникационных сетях трафик для того чтобы испытывать минимальные задержки должен проходить минимальное количество узлов и иметь минимальное время доставки, однако на практике такое реализуется лишь в магистральных сетях, в остальных случаях пропускная способность узлов, ребер, промежуточных буферов, а также оконечного оборудования может быть неравномерна и сильно ограничена.

Таким образом, на ресурсной сети не все пути является допустимыми. В работе исследованы модели графов сетей с ограничениями на достижимость, а также введены и изучены сети с распределением двух ресурсов.

В связи с развитием инфокоммуникационных сетей теория графов была значительно расширена, их функционирование уже невозможно представить без использования основных алгоритмов теории графов, в частности алгоритма нахождения кратчайшего пути между двумя

[1] Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 19-07-00525 А).

вершинами графа. С одной стороны, появление разнородного оборудования на сетях привело к тому, что сети стали неоднородные, а с другой постоянно растёт потребность к увеличению скорости передачи. Объединяют эти задачи общая математическая модель – ориентированный граф, лежащая в их основе, и алгоритмы, которые используются для решения этих задач.

Любую сеть передачи данных можно представить как ориентированный граф, каждая дуга которого – "труба" между соответствующими вершинами графа, пропускающая через себя "вещество" только в одном направлении – от начальной вершины в конечную вершину. Каждой дуге соответствует положительное число, называемое её пропускной способностью ("площадь поперечного сечения трубы").

Ориентированные графы с нестандартной достижимостью являются орграфами, в которых множество допустимых путей (т.е. возможных для рассмотрения) не совпадает со своим множеством путей на графе. Допустимые пути определяются некоторыми дополнительными условиями на их формирование, называемыми типом ограничений на достижимость. Классическим примером графов с ограничениями на достижимость являются графы со смешанной достижимостью. В таких графах имеется выделенное подмножество дуг, по которым смешанный путь не имеет права проходить подряд. Наличие ограничений на достижимость существенно меняет классическую ситуацию, когда наличие пути из "первой вершины" во "вторую вершину", а из "второй вершины" в "третью вершину" означает и наличие пути из "первой вершины" в "третью вершину". В случае нестандартной достижимости это свойство, называемое транзитивностью "множества путей" уже, как правило, пропадает, что влечет за собой необходимость разработки новых алгоритмов нахождения кратчайших путей, допустимых потоков и т.п.

Ориентированным графом называют тройку G(X,U,f), где X – множество вершин графа и не равно нулю, U – множество дуг графа, $f : X \to X \times X$ – отображение смежности (инцидентности). Вершины множества X_R будем называть нейтральными, а множества X_Z – запрещенными.

Ограничения на достижимость на графе могут быть не строгого и строгого типов.

К ограничениям достижимости нестрогого типа относятся графы с накоплением неубывающей **магнитности**, графы с накоплением-исчезновением магнитности, графы с условием биполярной магнитности и графы с условием механической достижимости.

Граф G(X,U,f) с магнитной достижимостью описывается множеством магнитных и не магнитных вершин $X = X_M \cup X_H$.

Путь μ называется **магнитно-накопительным путем** порядка k длины n с неубывающей магнитностью на графе G, если выполняется условие:

$$\forall m(\lambda_\mu(m) \geq k)(\theta^+((p_2 \circ f \circ \mu)(m) \cap U_M \neq \emptyset) \Rightarrow (\mu(m+1) \in U_M)$$

где $\lambda_\mu(j) = \sum_{m=1}^{j} |\mu(m) \cap U_m|$ – числовая характеристика отрезка пути.

Другими словами, путь на графе является допустимым при выполнении следующим условия: если характеристика λ_μ не меньше величины k и есть исходящие магнитные дуги, то следующая дуга пути должна быть магнитной.

К ограничениям достижимости строгого типа относятся графы с условием вентильной достижимости, графы с барьерной достижимостью, графы с ограниченной магнитной достижимостью и графы с ограниченной монотонной достижимостью.

Пусть G(X,U,f) – граф, X= $X_O \cup X_+ \cup X_B$, множества X_O, X_+ и X_B попарно не пересекаются. Множество X_O – множество нейтральных вершин, X_+ – множеством вершин, увеличивающих барьерный показатель (множество увеличивающих вершин), X_B – множество барьерных вершин.

Тогда каждым отрезком $[0;i]_z$ (где $0 \le i \le n$) пути $\tilde{x} = \{x_i\}_{i=0}^n$ свяжем числовую характеристику $\beta_{\tilde{x}}(i)$, определенную индуктивно следующим:

1. $\beta_{\tilde{x}}(0) = \begin{cases} 1, \ \textit{если } x_0 \in X_+ \\ 0, \ \ \textit{противном случае} \end{cases}$

2. $\beta_{\tilde{x}}(i+1) = \begin{cases} \beta_{\tilde{x}}(i), \ \textit{если } x_i \in X_N \\ \beta_{\tilde{x}}(i)+1, \ \textit{если } x_i \in X_+ \\ 0, \ \textit{если } x_i \in X_B \end{cases}$

Характеристика $\beta_{\tilde{x}}(i)$ называется величиной накопленной энергии на отрезке $[0;i]_z$ пути $\tilde{x} = \{x_i\}_{i=0}^n$.

Другими словами, если к m-му шагу путь $\tilde{x} = \{x_i\}_{i=0}^n$ от своего начала накопил величину барьерного показателя $\beta_{\tilde{x}}(m-1)$ большую либо равную h, то на последующем шаге становятся допустимыми для прохождения вершины из множества X_B.

В данной работе мы будем рассматривать ресурсные сети с различными типами ограничений на достижимость, поскольку процесс функционирования ресурсной сети, если его рассматривать не потактно, а разворачивающимся во времени можно считать процессом передачи ресурса по путям. В случае ресурсных сетей с ограничениями на достижимость эти пути будут рассматриваться только из множества допустимых, определенных типом достижимости, т.е. нестандартная достижимость понимается нами как некоторое ограничение на множества путей ресурсной сети.

2. СТРУКТУРА ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

Программа имитационного моделирования разработана на языке Phyton и состоит из трех модулей: вычислительного calc.py, графического drawing.py и главного main.py.

В вычислительном модуле происходит расчет путей на предварительно заданном графе. Данной граф задается с помощью матрицы смежности в csv формате. В графическом модуле генерируется изображение с заданным графом и всеми допустимыми путями на нем. С помощью модуля main.py происходит инициализация программы. Блок-схема созданной программы, представленной на рисунке 1.

3. ГРАФИЧЕСКИЙ МОДУЛЬ

3.1. Вспомогательные функции

Среди множества вспомогательных функций важную роль функция generate_pos, которая создает массив координат вершин графа. На вход функция принимает граф в виде словаря, выходными данными является словарь с записями вида {V : [x, y]}, где V – вершина, а x и y – ее координаты по осям X и Y соответственно

def generate_pos(g)**:**

Так как в функции происходит расчет координат вершин, требуется выбрать систему координат, которая в дальнейшем будет использоваться в алгоритмах networkx. Была выбрана связанная с данными (transData) система координат matplotlib. Функция задает позиции вершин так, чтобы минимальное количество ребер между вершинами пересекали другие вершины или проходили вблизи. Эмпирическим путем было найдено оптимальное правило расположения вершин на плоскости. Алгоритм основывается на присвоении вершине координат в зависимости от ее номера. Вершине 1 всегда присваивается координата (0, 3). По оси X координаты вершины вычислялись по формуле (i+1) /2, то есть на одной координате X находятся по две вершины, четная и следующая после нее нечетная. У следующей четной координата X уже больше. По оси Y порядок расположения более сложный. От первой вершины откладываются

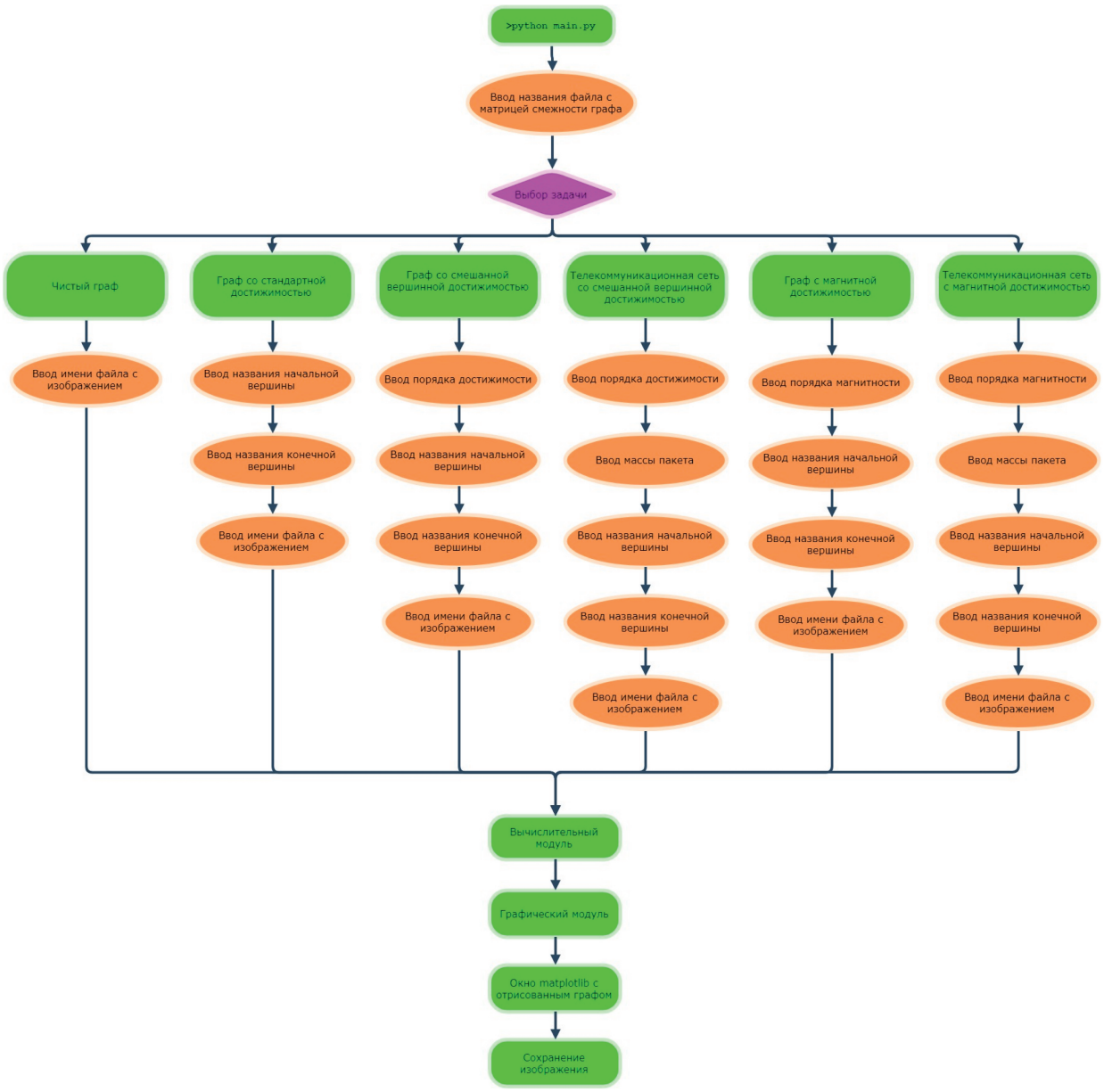

Рис. 1. Универсальная блок-схема программы.

две воображаемые параболы, симметричные относительно горизонтальной линии, проходящей через первую вершину (то есть прямую y = 3). Вершины располагаются так, что первая, третья, пятая и так далее четная вершина находится выше (на 3 единицы) над верхней параболой, а каждая вторая, четвертая, шестая и так далее нечетная вершина – ниже этой параболы (на ту же самую величину). Аналогично с вершинами с нечетными номерами, только они осциллируют около нижней параболы.

3.2. Алгоритм отрисовки чистого графа

Все функции рисования в программе построены на принципиально одинаковом базисе, на котором надстроены различные элементы кода, меняющий алгоритм в зависимости от задачи. Для рисования используется библиотека networkx

def simple_draw_digraph(digraph, digraph_name, ipos, inc_nodes = [], dec_nodes = []):

На вход функция получает вектор аргументов digraph, digraph_name, ipos, inc_nodes и dec_nodes. В первом содержится сам рисуемые граф, во втором содержится название, с которым будет сохраняться png изображение нарисованного файла. Параметр ipos – это флаг, который отвечает за тип расположения вершин. Если он равен 1, то будет выбран описанный выше тип, если иное значение – один из предустановленных в библиотеке networkx. В последних двух могут содержаться массивы увеличивающих и уменьшающих вершин, если таковые заданы на графе. По умолчанию эти параметры равны пустым спискам.

4. ВЫЧИСЛИТЕЛЬНЫЙ МОДУЛЬ

Для построения имитационного моделирования использовался язык Python. Для извлечения необходимой информации из текстового файла заданной матрицы смежности графа была использована функция extract, которая предназначена для извлечения из текстового файла ранее созданного графа, а также сопутствующей информации (списки вершин с условиями), если она есть. Для работы с файлами используется библиотека pandas. Возвращаемые данные – словарь Python.

5. АЛГОРИТМ ПОИСКА ПУТЕЙ В ГРАФЕ СО СТАНДАРТНОЙ ДОСТИЖИМОСТЬЮ

В основу алгоритмов поиска путей был взят алгоритм поиска по графу в глубину (Deep-First-Search, DFS), так как в нем легко реализуется добавление новых видов достижимости и весов вершин

def dfs**(**graph**,** start**,** goal**):** (1)

На вход функции (1) подаются ранее заданный граф, начальная и конечная вершины пути.

Работу алгоритма поиска в глубину можно сравнить с распространением волны огня по дереву, только вместо дерева здесь граф. Огонь начинается на стартовой вершине, и за каждый проход цикла в теле функции он распространяется на расстояние одного ребра по графу во все возможные ответвления. И так, пока все пути не приведут в целевую вершину или не погаснут в тупиках.

Рассмотрим пример программной задачи – граф. Каждой вершине, заданной через одинарные кавычки, соответствует список ее соседей, заданных через квадратные скобки. Это позволяет задавать в одном формате как ориентированные, так и не ориентированные графы.

Пример 1. Пример задания графа

graph = **{**’1’**: [**’2’**,** ’3’**],**
’2’**: [**’4’**,** ’9’**],**
’3’**: [**’5’**],**
’4’**: [**’6’**],**
’5’**: [**’6’**,** ’7’**],**
’6’**: [**’8’**],**
’7’**: [**’9’**],**
’8’**: [**’3’**,**’7’**,**’10’**],**
’9’**: [**’10’**],**
’10’**: []}**

В отдельной переменной stack хранится “фронт распространения” алгоритма, его текущее положение на графе. По сути это список кортежей, где первый элемент – последняя вершина пути, а второй – сам путь. Распространение фронта поиска идет по циклу while до момента, пока фронт (переменная stack) не станет пустым. Функция возвращает генератор Python, элементы которого – пути, соединяющие две заданные вершины.

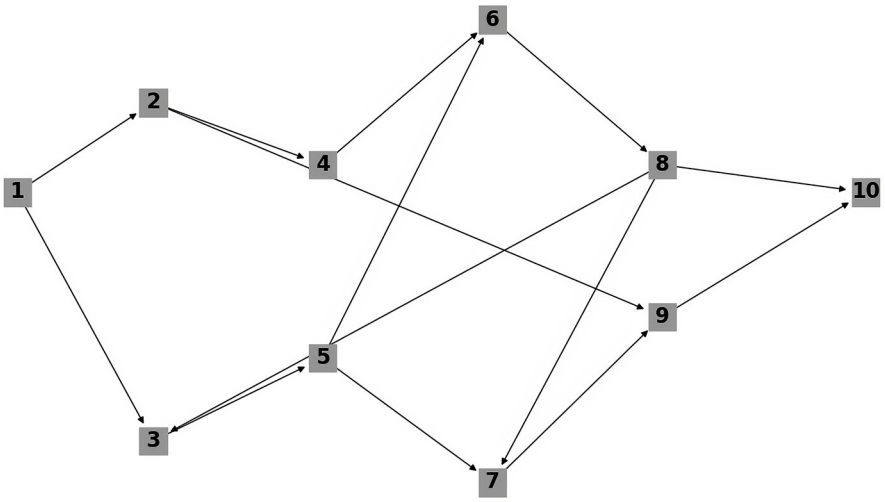

Рис. 2. Отображение заданного графа.

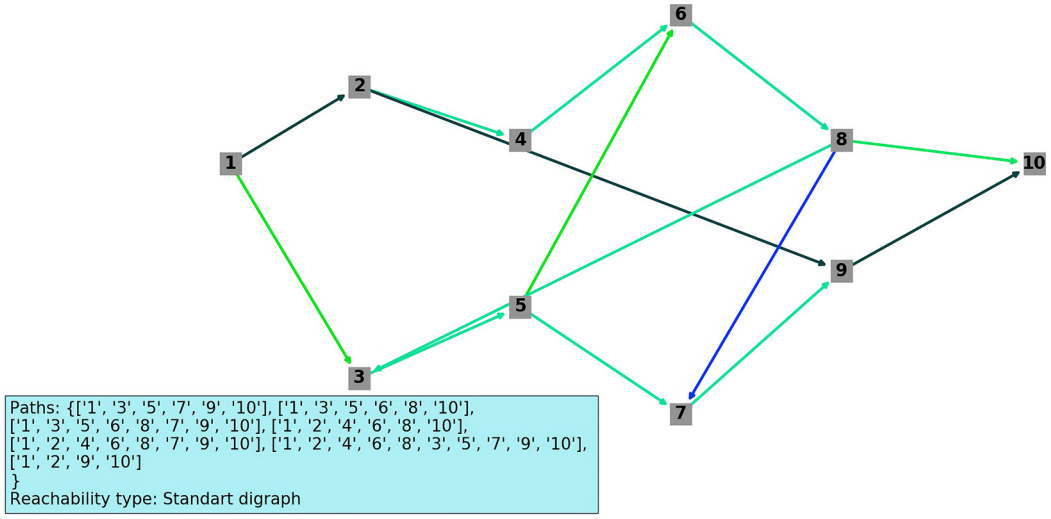

Рис. 3. Отображение пути в графе со стандартной достижимостью.

5.1. Алгоритм поиска путей в графе с барьерной достижимостью

Чтобы алгоритм DFS стал учитывать смешанную вершинную (барьерную) достижимость заданного порядка k (Vertex Mixed Reachability of k degree, VMRk), был дополнен предыдущий алгоритм

def dfs_vmrk**(**graph**,** inc_nodes**,** k**,** start**,** goal**):** (2)

В алгоритме по расчету пути на графе с барьерной на вход функции (2) к параметрам из функции (1) добавляются: множество увеличивающих вершин и уровень барьера, или порядок достижимости.

Теперь в переменной stack кортежи, помимо последней вершины и пути, содержат счетчик i, который уникален для каждого кортежа, то есть для каждого текущего пути фронта распространения алгоритма. При переходе на следующую вершину, если она содержится в списке увеличивающих, то счетчик увеличивается на единицу. Переход может быть осуществлен, только если соседняя вершина не содержится в списке увеличивающих вершин и при

этом счетчик меньше барьера уровня k. Если переход происходит на нейтральную вершину, счетчик обнуляется.

Пример на графе из 18 вершин, пути из вершины 12 в 17 при k=4 представлен на рисунке 4. Зеленые вершины являются увеличивающими.

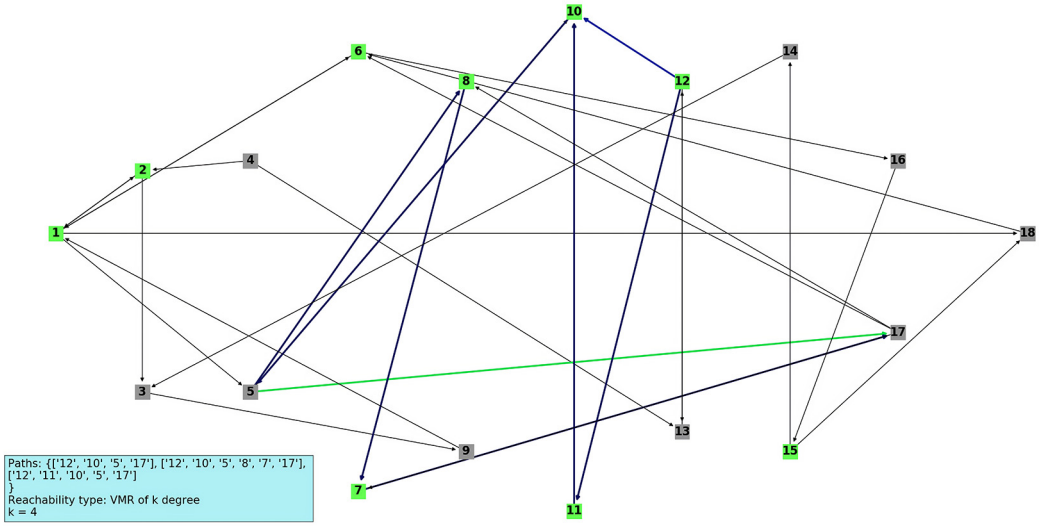

Рис. 4. Отображение пути в графе с барьерной достижимостью.

5.2. Алгоритм поиска путей во взвешенном графе (модели телекоммуникационной сети) с барьерной достижимостью

Модель телекоммуникационной сети реализована с помощью взвешенного графа и расчета времени прохода пакета по некоторому пути на произвольном графе

def dfs_vmrk_nets(graph, inc_nodes, k, start, goal, m): (3)

В алгоритме по расчету пути на взвешенном графе с барьерной достижимостью на вход функции (3) к параметрам из функции (2) добавляется вес пакета m. Во взвешенном графе задание вершин, их соседей и весов выглядит следующим образом:

Пример 2. Задание ориентированного графа

graph = **{**'1'**: {**'2'**:45,** '3'**:40},**
'2'**: {**'4'**:15,** '9'**:35},**
'3'**: {**'5'**:45},**
'4'**: {**'6'**:20},**
'5'**: {**'6'**:10,** '7'**:25},**
'6'**: {**'8'**:40},**
'7'**: {**'9'**:30},**
'8'**: {**'3'**:50,** '7'**:35,** '10'**:5},**
'9'**: {**'10'**:25},**
'10'**:{}}**

Появляется новая переменная t, которая обозначает время, затрачиваемое пакетом на проход по пути, содержащемся в данном кортеже. Она, как и счетчик i, уникальна для каждого кортежа.

При переходе к следующей вершине к переменной времени t каждого пути прибавляется величина m/v, где v – вес на ребре, соединяющим текущую вершину (последнюю в текущем пути) и следующую. Так как следующих вершин может быть несколько, то для того, чтобы время прохождения каждого пути не прибавлялось при переборе следующих вершин, производится операция вычитания из t величины m/v, как описано в примере 3.

Пример 3. Пусть взят участок графа, как представлено на рисунке 5. Пусть алгоритм находится на вершине 3.

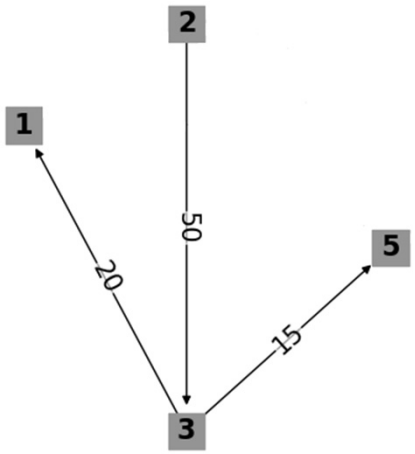

Рис. 5. Участок взвешенного графа.

Это значит, что при весе пакета m = 10 (к примеру) текущее время пути составляет 5 (50/10). Из третьей мы сможем перейти как на 1, так и на 5 вершину. Пусть обход доступных вершин начался с 1. Тогда к t прибавляется 2 и в переменную stack отправляется новый элемент списка с временем 5+(20/10)=7. И если мы не отбавим от t это время прохода по пути из 3 в 1, то получим, что общее время пути 2-3-4 составит 7+(15/10)=8.5. Тогда как очевидно, что время перехода 3-1 учитывать нельзя. Поэтому из t каждый раз надо вычитать m/v после прибавления и отправки в переменную stack.

Пример 4. Поиск самого быстрого пути из вершины 6 в вершину 11 порядка k=1:

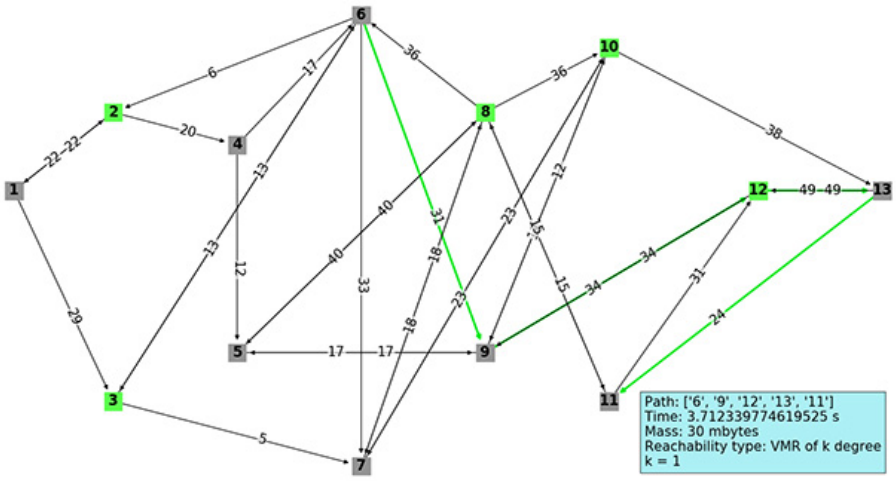

Рис. 6. Отображение пути во взвешенном графе с барьерной достижимостью.

Быстрейший путь исходя из заданных начальных условий (см рисунок 6) будет 6-9-12-13-11. Вес файла 30 Мбит. Время прохождения по этому пути составила

$$\frac{30}{31} + \frac{30}{34} + \frac{30}{49} + \frac{30}{24} = 3.71.$$

5.3. Алгоритм поиска путей в графе с магнитной достижимостью

Алгоритм магнитной достижимости порядка k (Magnetic Reachability of k degree, MnRk) незначительно отличается от алгоритма поиска в пути в графе со смешанной вершинной достижимостью

def dfs_mnrk **(**graph**,** inc_nodes**,** k**,** start**,** goal**,** dec_nodes**):** (4)

В алгоритме по расчету пути на графе с барьерной достижимостью на вход функции (4) к параметрам из функции (2) добавляется еще один вид вершин – уменьшающих счетчик. В данном алгоритме отсутствует ограничение на переход к какой-либо вершине, условие достижимости появляется только при обнаружении целевой вершины.

Если при обходе соседей некоторой текущей вершины алгоритм встретил целевую, то сначала он решает, какая операция будет проведена со счетчиком: увеличить его, если целевая вершина в увеличивающих – происходит прибавление единицы, уменьшить его на единицу, если целевая вершина в уменьшающих.

При условии, что счетчик равен магнитности k, путь добавляется в генератор.

Пример поиска путей на графе при магнитности k = 1 из вершины 11 в 9:

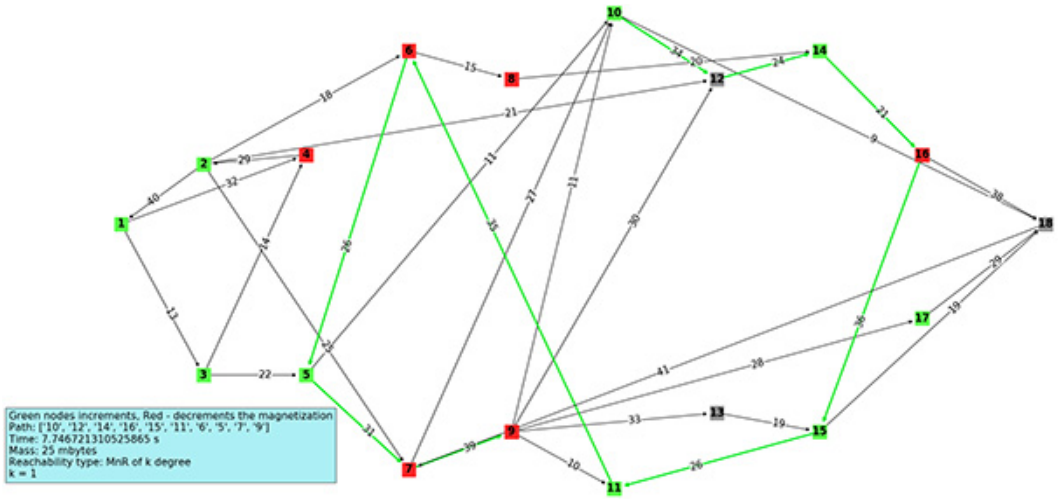

Рис. 7. Отображение пути в графе с магнитной достижимостью.

На графе, отображенном на рисунке 7, зеленым обозначены вершины из списка увеличивающих магнитность пути, а красным – из списка уменьшающих. Серые – нейтральные вершины. Путь оказался единственным: 11-6-5-10-12-14-16-15-18-7-9.

5.4. Алгоритм поиска путей во взвешенном графе (модели телекоммуникационной сети) с магнитной достижимостью

Этот алгоритм сочетает в себе MnRk и те же самые правила расчета времени, что и в VMRk для телекоммуникационных сетей. Отличие от MnRk заключается в наличии в нем нового параметра m, в который определяет вес (размер) пакета

def dfs_mnrk_tn(graph, inc_nodes, k, start, goal, m, dec_nodes = **[]**, check = **0):** (5)

В алгоритме по расчету пути на взвешенном графе с магнитной достижимостью на вход функции (5) к параметрам из функции (3) добавляется переменная под список вершин, уменьшающих счетчик и флаг, как параметр функции, который нужен для удобства работы с алгоритмом. Когда флаг:

- 0, функция выберет самый быстрый путь из удовлетворяющих заданной магнитности k.
- 1, то функция составит список возможных путей с их порядками магнитности, и вернет самый быстрый из них (то есть сам быстрый путь между заданными вершинами).
- 2, то вернется словарь, в котором ключами будут являться порядки магнитности путей, по котором можно добраться из первой заданной вершины во вторую, а значениями – количество путей для каждого порядка магнитности.

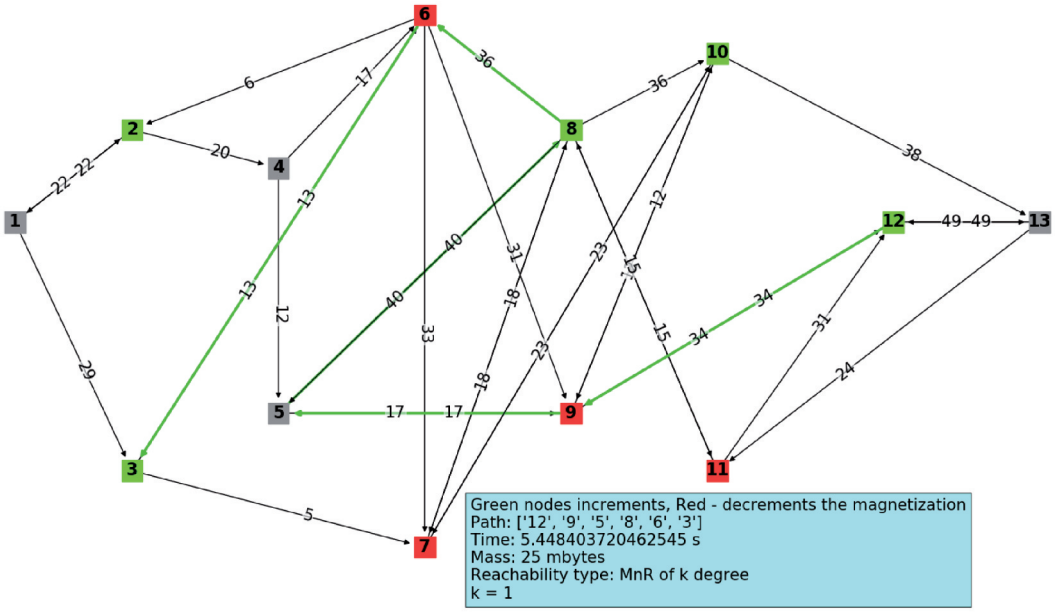

Рис. 8. Отображение пути во взвешенном графе с магнитной достижимостью.

Быстрейший путь исходя из заданных начальных условий (см рисунок 8) будет 12-9-5-8-6-3. Вес файла 25 Мбит. Время прохождения по этому пути составила

$$\frac{25}{34}+\frac{25}{17}+\frac{25}{40}+\frac{25}{36}+\frac{25}{13}=5.45.$$

В дальнейшем планируется рассмотрение совместной задачи моделирования на ограничение графа по его вершинам и ребрам, что будет более точно соответствовать моделям современных сетей передачи данных.

Литература

1. Абдулрахман, Х. Ресурсные сети с вентильной достижимостью / Х. Абдулрахман, Я.М. Ерусалимский, В.А. Скороходов // Инженерный вестник Дона. – 2018. – № 4. – 12 с. URL: www.ivdon.ru/magazine/archive/ n4y2018/5264.

2. Абдулрахман, Х. Полные двух ресурсные сети с петлями / Х. Абдулрахман, В.А. Скороходов // Известия ВУЗов. Северо-Кавказский регион. Естественные науки. – 2016. – № 2. – С. 10–16.

3. Абдулрахман, Х. Ресурсные сети с магнитной достижимостью / Х. Абдулрахман, В.А. Скороходов // Известия ВУЗов. Северо-Кавказский регион. Естественные науки. – 2016. – № 4. – С. 4–10.

4. Скороходов, В.А. Динамические ресурсные сети. Случай малого ресурса / В.А. Скороходов, Х. Абдулрахман // Вестник Воронежского государственного университета. Серия: Физика. Математика. – 2018. – № 4. – С.186-194.

5. Антонова В.М., Богомолова Н.Е./Изучение особенностей конфигурирования Mesh – сетей в Matlab/"Фундаментальные проблемы радиоэлектронного приборостроения", труды международной научно-технической конференции, 20-24 ноября 2017, МИРЭА, "INTERMATIC-2017", М. 2017

Simulation of graphs with different types of reachability in the Python program

V.M. Antonova, B.M. Zakhir, N.A. Kuznetsov

The aim of the article is to study various resource networks with various reachability restrictions.

KEYWORDS: graphs, flows on graphs, magnetic reachability, Barrier reachability.