УДК 004.021

Разработка программы нахождения кратчайшего пути в графе с помощью муравьиного алгоритма

Скандаленко Александр Валерьевич
Приамурский государственный университет им. Шолом-Алейхема
студент

Баженов Руслан Иванович
Приамурский государственный университет им. Шолом-Алейхема
к.п.н., доцент, зав. кафедрой информационных систем, математики и правовой информатики

Аннотация
В данной статье рассмотрена проблема создания программ для нахождения пути в графе. В качестве исследования были изучены известные решения по данному вопросу, необходимые языки программирования и необходимое программное обеспечение. В результате была разработана программа по нахождению кратчайшего пути, использующая муравьиный алгоритм.
Ключевые слова: муравьиный алгоритм, алгоритмы поиска пути, граф

Development of a program for finding the shortest path in a graph using ant algorithm

Skandalenko Aleksandr Valerievich
Sholom-Aleichem Priamursky State University
student

Bazhenov Ruslan Ivanovich
Sholom-Aleichem Priamursky State University
Candidate of pedagogical sciences, associate professor, Head of the Department of Information Systems, Mathematics and Legal Informatics

Abstract
This article discusses the problem of creating programs to find the path in the graph. The well-known solutions on this issue, the necessary programming languages and the necessary software are studied. For example, the C++ programming language and the Dev C++program were considered.
Keywords: ant algorithm, pathfinding algorithms, graph

В реальной жизни есть множество задач с нахождением кратчайшего пути среди множества разных путей. Lанную задачу пытались решать математики, однако, в реальных условиях данная задача имеет огромное множество данных, для которых потребовались машинные вычислительные

мощности. Именно для таких целей и были придуманы различные алгоритмы поиска кратчайшего пути, среди которых есть муравьиный алгоритм.

Цель данного исследования заключается в создании с помощью языка программирования C++ программы, способной найти кратчайший путь между вершинами в графе на основе муравьиного алгоритма.

В статье А.А. Дроздова и Р.И. Баженова был рассмотрен вариант применения муравьиного алгоритма для решения задачи коммивояджера в программе MathLab [1]. М.И. Кирсанов подробно описал вариант применения данного алгоритма для распределения агентов по рёбрам взвешенного графа [2]. Ю.И. Нечаев и О.Н. Петров предложили использование муравьиного алгоритма в современной теории катастроф [3]. А.Д. Данилов и В.А. Ломакин нашли применение муравьиному алгоритму как инструменту для наиболее эффективного распределения заданий между различными станками [4]. О.С. Нургаянова и И.И. Янбаев подняли вопрос о модификациях муравьиного алгоритма для более эффективного решения задачи Коммивояджера [5]. Д.А. Мухина использовала муравьиный алгоритм для маршрутизации транспорта [6]. В.С. Мельник исследовал муравьиный алгоритм для маршрутизации пакетов в компьютерных сетях [7].

Муравьиный алгоритм основан на способе поиска пищи у реальных муравьёв, который выглядит следующим образом:

1. Группа муравьёв-разведчиков ищет пищу. Найдя её, они помечают пути обратно своими феромонами. Таким образом, получается несколько путей.
2. Группа муравьёв, отправленная по следу феромонов муравьёв-разведчиков, укрепляет тропы феромонов, однако, т.к. феромоны имеют свойство испаряться, то тропа, по которой муравей будет долго добираться до пищи, будет менее приоритетной, чем тропа, где муравей доберётся до пищи быстрее.
3. Муравьи-рабочие начинают ходить по полученной тропе до пищи и обратно и укреплять феромонами тропу. Некоторые муравьи будут периодически отходить от проложенной тропы в поисках более оптимального маршрута и, если муравей находит такой маршрут, все последующие муравьи будут ходить уже по новому маршруту.

Таким образом муравьи находят наиболее оптимальный маршрут среди множества путей.

Муравей, феромоны и ветер, испаряющий феромоны, в муравьином алгоритме – это абстрактные понятия и переменные. Сам муравьиный алгоритм выглядит следующим образом:

1. «Муравей» ищет путь до цели – конечной вершины графа.
2. Если «муравей» находит конечную вершину или вершину, помеченную феромонами, то «муравей» помечает обратный путь – ребро, по которому прошёл муравей, количеством феромонов, равному количеству феромонов конечной вершины, умноженному на константу силы муравьиных феромонов.

3. Однако, есть вероятность того, что муравей выберет другой путь, на котором феромонов может не быть. Вероятность выбора другого пути обратно пропорциональна силе феромонов на всех рёбрах, направленных из вершины. Вероятность прохождения по пути высчитывается по формуле: $P_i = \frac{l_i^q * f_i^p}{\sum_{k=0}^{N} l_k^q * f_k^p}$, где P_i – вероятность перехода по i-тому пути, l – величина, обратная весу ребра, f – количество феромона, q – сила «ветра» или обратная величина силы «феромонов», p – сила «феромонов».

4. Если «муравей» зашёл в тупик, т.е. вершину без выходящих рёбер, «муравей» возвращается в стартовую вершину. Также, «муравей» возвращается в стартовую вершину после прибытия в финишную вершину.

5. После выполнения алгоритма для нескольких итераций «муравьёв», выполняется операция «дуновение ветра», что представляет из себя умножение силы феромонов каждого ребра на константу, обратно пропорциональную силе феромонов (1 минус сила феромонов), таким образом, чем сильнее феромоны, тем сильнее они испаряются и тем чаще муравьи будут искать другой путь и реже заходить на уже проложенные тропы.

6. Количество феромонов на каждой вершине приравнивается к наибольшему количеству феромонов на рёбрах, выходящих из вершины.

7. У каждого «муравья» есть предел количества шагов. Это предусмотрено для того, чтобы «муравей» не блуждал в бесконечном цикле.

Первая версия алгоритма была предложена Марко Дориго в 1992 году. Со временем, у данного алгоритма появились вариации:

- Элитарная муравьиная система, в которой из нескольких муравьёв выделяется группа так называемой элиты. Эта группа, проходя по маршруту, укрепляет его сильнее остальных муравьёв.
- MMAS (Max-Min Ant System) – это муравьиный алгоритм с ограничением на количества феромонов на пути.
- Ранговая муравьиная система, где все решения имеют значимость и для каждого решения выделяется своё значение феромонов, таким образом более подходящие решения будут получать больше феромона, чем менее подходящие.

На сегодняшний день самая популярная задача, для решения которой потребуется муравьиный алгоритм, эта задача с нахождением пути для автотранспорта. Именно такая задача будет в примере.

В данном исследовании будет произведено создание с помощью языка C++ программы, использующей муравьиный алгоритм. Для создания такой программы был использован компилятор Dev-C++. В результате работы данного компилятора, была создана консольная программа, работающая с 2 текстовыми файлами (рис. 1)

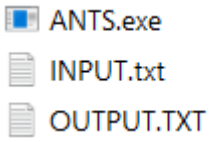

Рисунок 1. Список файлов (программа и 2 текстовых файла)

Для работы программы, необходимо заполнить текстовый файл INPUT.txt по следующему образцу:

1. Первая строчка заполняется числами в следующем порядке:
   1.1. Номер начальной вершины
   1.2. Номер конечной вершины
   1.3. Количество рёбер
   1.4. Количество вершин
   1.5. Сила феромонов. Записывается как число в пределе от 0 до 1.
2. Последующие строки в количестве равном числу рёбер заполняются в следующем формате:
   2.1. Номер вершины, из которой выходит ребро
   2.2. Номер вершины, куда входит ребро
   2.3. Вес этого ребра. Записывается как число в пределе от 0 до 1.

Для примера можно взять граф на рисунке 2.

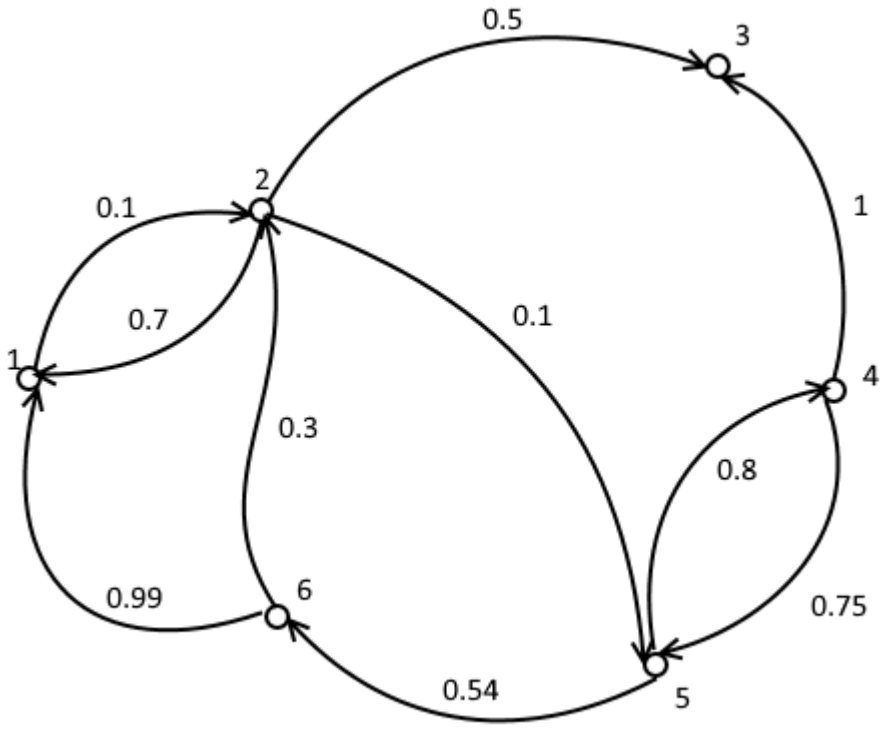

Рисунок 2. Граф, используемый в примере

В первом примере, возьмём в качестве начальной вершины вершину №1, а в качестве конечной – вершину № 6. Силу муравьиных феромонов установим на значение в 0.7

В результате, после переноса графа в текстовый формат, в INPUT.txt были записаны строки на рисунке 3.

```
1 6 10 6 0.7
1 2 0.1
2 1 0.7
2 3 0.5
2 5 0.1
4 3 1
4 5 0.75
5 4 0.8
5 6 0.54
6 1 0.99
6 2 0.3
```

Рисунок 3. Пример формата ввода данных в INPUT.txt

После этого, нужно запустить программу ANTS.exe. У программы нет интерфейса, поэтому она закроется сразу после выполнения задачи. После выполнения данной задачи, в файле OUTPUT.txt появятся строки как на рисунке 4.

```
1 2 0.1 # 0.343 * 100%
2 1 0.7 # 0.02 * 3.92157%
2 3 0.5 # 0 * 0%
2 5 0.1 # 0.49 * 96.0784%
4 3 1 # 0 * 0%
4 5 0.75 # 0.49 * 100%
5 4 0.8 # 0.1029 * 12.816%
5 6 0.54 # 0.7 * 87.184%
6 1 0.99 # 0 * 0%
6 2 0.3 # 0 * 0%
```

Рисунок 4. Пример формата выходных данных из OUTPUT.txt

Как можно заметить, программа создала строки с распределением «муравьёв» по ребрам. Данные строки оформлены по следующему формату:

1. Каждая строка – ребро.
2. Первое число в строке – номер вершины, из которой выходит ребро.
3. Второе число – номер вершины, в которую входит ребро.
4. Решётка – знак-разделитель. После него обозначена сила феромонов на данном ребре.
5. Знак множителя – знак-разделитель. После него обозначено процентное отношение муравьёв, выходящих из вершины, из которой начинается ребро, и выбравших этот маршрут.

Для удобства, перенесём эти данные на визуальный граф (рис. 5)

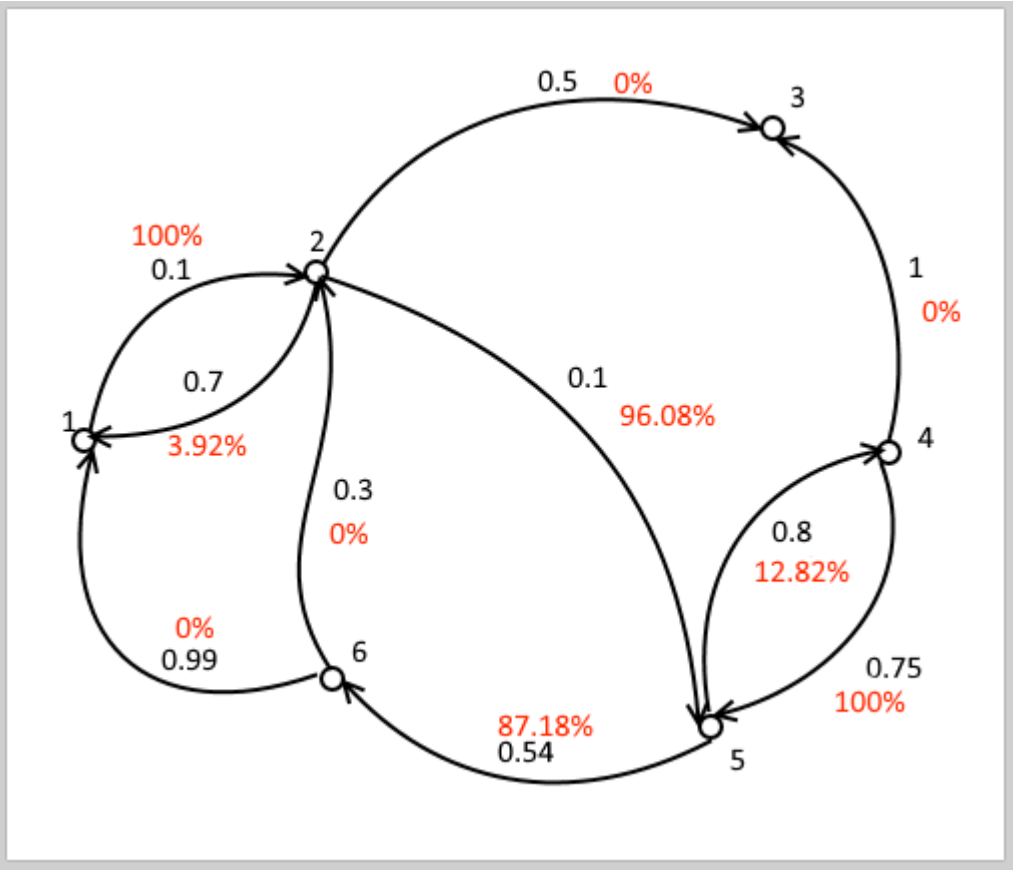

Рисунок 5. Графическое отображение выходных данных. Проценты выделены другим цветом

В данном случае видно, что все «муравьи», попавшие в вершины 1 и 4 идут по единственному возможному пути. Часть муравьёв возвращается обратно (ребро 2-1), а также, часть муравьёв уходит в маршрут, из которого все возвращаются назад (ребро 5-4). Однако, большинство «муравьёв» прошло по маршруту 1-2-5-6.

Для другого примера, воспользуемся этим же графом, но точку начала расположим в вершине 6, а конечную – в вершине 1. Результат показан на рисунках 6 и 7.

```
1 2 0.1 # 0 * 0%
2 1 0.7 # 0.7 * 100%
2 3 0.5 # 0 * 0%
2 5 0.1 # 0 * 0%
4 3 1 # 0 * 0%
4 5 0.75 # 0 * 0%
5 4 0.8 # 0 * 0%
5 6 0.54 # 0 * 0%
6 1 0.99 # 0.21 * 30%
6 2 0.3 # 0.49 * 70%
```

Рисунок 6. Пример выходных данных для примера №2

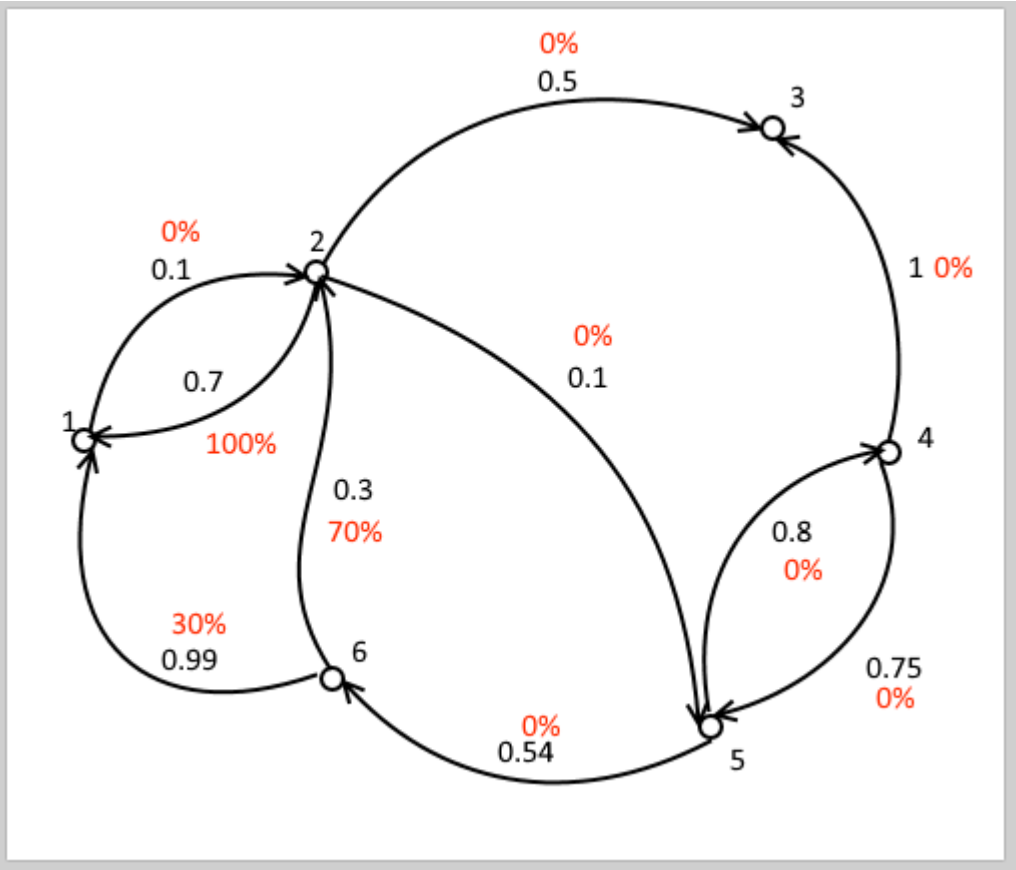

Рисунок 7. Графическое отображение выходных данных для примера №2

Как можно заметить, данный алгоритм построил уже не маршрут, а распределение сил. Из начальной вершины 30% выбрали самый неудобный маршрут, в то время как остальные 70 выбрали более лёгкий, но более длинный маршрут. Однако, если заметить, можно увидеть, что вес обоих маршрутов примерно одинаков: маршрут 6-1 имеет вес 0.99, а маршрут 6-2-1 имеет вес 0.7+0.3=1. Это объясняется тем, что в данном алгоритме муравьи видят только ближайшие вершины и не оценивают общую продолжительность пути.

Для третьего примера используем тот же граф, однако добавим ребро 7-3 с весом 0.1. В качестве стартовой вершины выберем вершину 1. Конечная вершина будет вершина 7.

Результат представлен на рисунках 8-10.

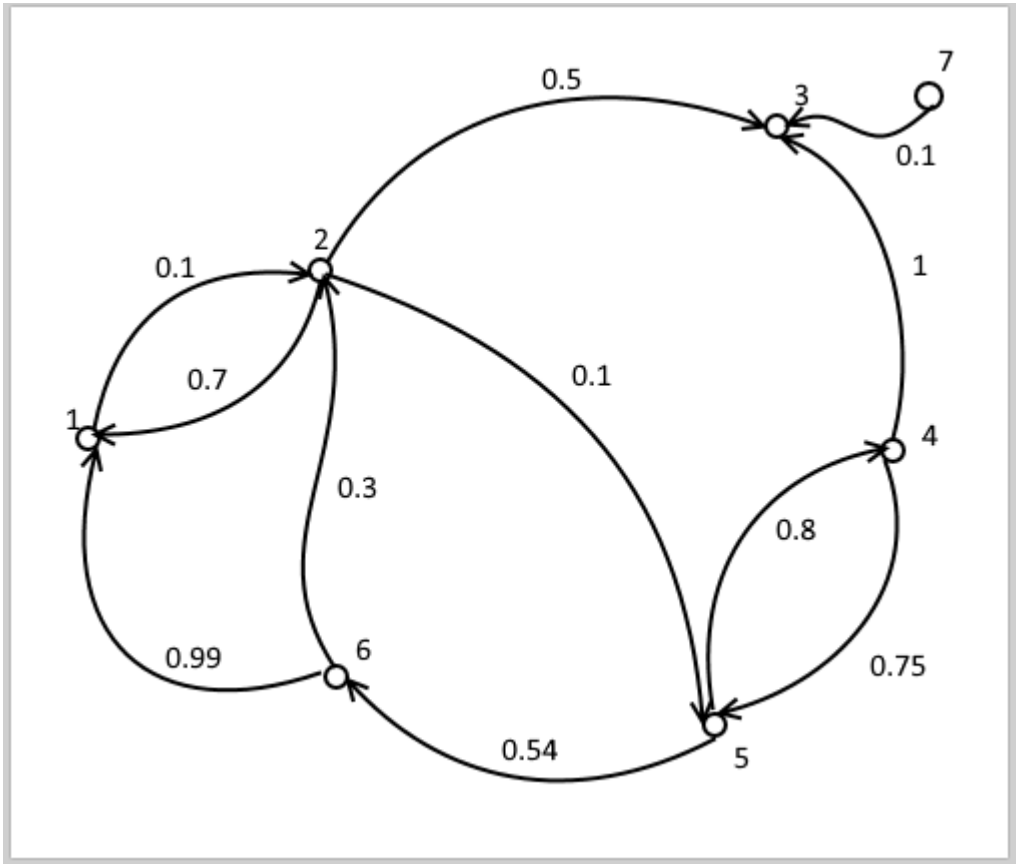

Рисунок 8. Графическое отображение графа для примера №3

```
1 2 0.1 # 0 * 0%
2 1 0.7 # 0 * 0%
2 3 0.5 # 0 * 0%
2 5 0.1 # 0 * 0%
4 3 1 # 0 * 0%
4 5 0.75 # 0 * 0%
5 4 0.8 # 0 * 0%
5 6 0.54 # 0 * 0%
6 1 0.99 # 0 * 0%
6 2 0.3 # 0 * 0%
7 3 0.1 # 0 * 0%
```

Рисунок 9. Пример выходных данных для примера №3

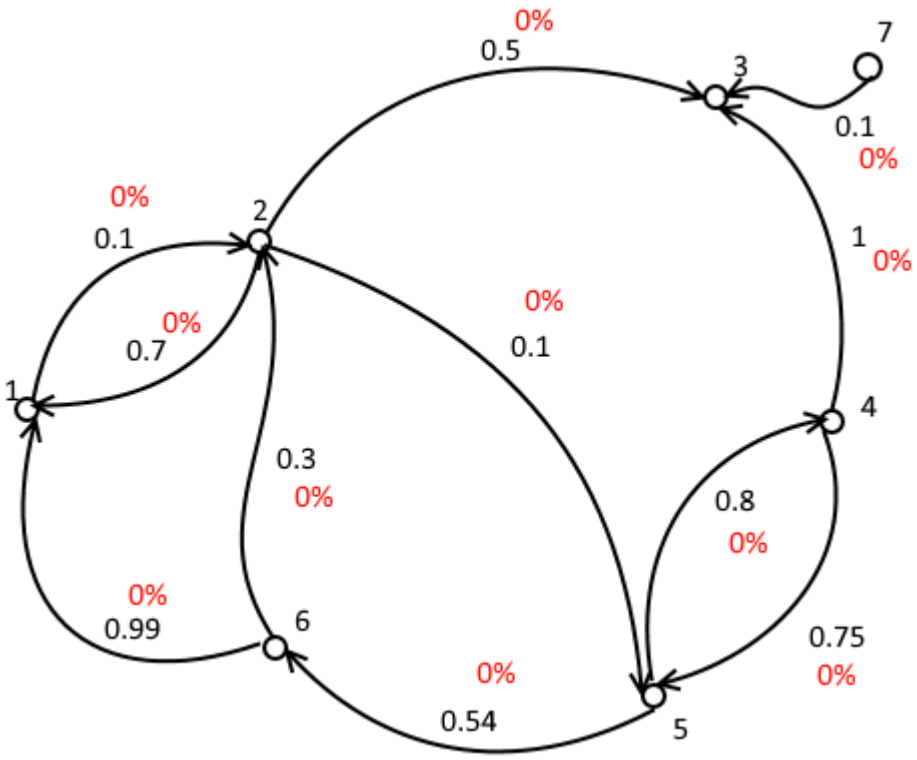

Рисунок 10. Графическое отображение данных для примера №3

Как можно заметить, в данном случае нет маршрута из вершины 1 в вершину 7.

Данная программа использует базовый муравьиный алгоритм и подходит как для ознакомления с данным алгоритмом, так и для симуляции реального трафика на реальных городских дорогах. В последнем случае, вес рёбер графа будет показывать процентную занятость дороги, т.е. насколько сильно дорога загружена: если вес равен 0, значит дорога полностью свободна, если вес равен 1, то по дороге двигаться невозможно, в значениях веса от 0 до 1, скорость транспорта будет снижена на указанное число.

Программы, использующие муравьиный алгоритм, не особо точны, т.к. есть более оптимальные алгоритмы поиска пути, однако муравьиный алгоритм является одним из самых простых и понятных алгоритмов для ознакомления с алгоритмами поиска пути.

Библиографический список

1. Дроздов А.А., Баженов Р.И. Нахождение оптимального пути с помощью муравьиного алгоритма в задаче коммивояджера // Постулат. 2018. №1 (27). С. 27.
2. Кирсанов М.И. Алгоритм разрешения конфликтов в задаче выбора оптимальных маршрутов группы агентов на плоскости // Постулат. 2017. №11 (25). С. 32.
3. Нечаев Ю.И., Петров О.Н. Моделирование самоорганизующейся системы на основе «муравьиного» алгоритма // Морские интеллектуальные

технологии. 2018. Т. 1. №2 (40). С. 153-159.
4. Данилов А.Д., Ломакин В.А. Применение муравьиного алгоритма в интеллектуальной системе диспетчеризации дискретного производства // Вестник Воронежского государственного технического университета. 2018. Т. 14. №4. С. 7-11.
5. Нургаянова О.С., Янбаев И.И. О модификации муравьиного алгоритма для решения задачи коммивояджера // В сборнике: Информационные технологии интеллектуальной поддержки принятия решений (ITIDS'2018) Труды VI Всероссийской конференции (с приглашением зарубежных ученых). 2018. С. 38-40.
6. Мухина Д.А. Муравьиный алгоритм в решении задачи маршрутизации транспорта // В сборнике: Информационные технологии и высокопроизводительные вычисления Материалы IV всероссийской научно-практической конференции. 2017. С. 91-103.
7. Мельник В.С. Исследование алгоритмов для решения задачи маршрутизации пакетов в компьютерной сети // Международная конференция по мягким вычислениям и измерениям. 2018. Т. 1. С. 673-676.