Государственное образовательное учреждение высшего профессионального

образования

«Московский государственный технический университет

имени Н. Э. Баумана»

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №7

По теме:

«Исследование муравьиного алгоритма»

Монахов Д. И.

ИУ7-54

Москва 2018

Оглавление

[1. Аналитическая часть 3](#_Toc532824976)

[1.1. Муравьиный алгоритм 3](#_Toc532824977)

[2. Конструкторская часть 5](#_Toc532824978)

[2.1 Разработка алгоритмов 5](#_Toc532824979)

[3. Технологическая часть 6](#_Toc532824980)

[3.1 Требования к программному обеспечению 6](#_Toc532824981)

[3.2 Средства реализации 6](#_Toc532824982)

[3.3 Листинг кода 7](#_Toc532824983)

[4. Экспериментальная часть 11](#_Toc532824984)

[4.1 Постановка эксперимента 11](#_Toc532824985)

[Заключение 20](#_Toc532824986)

# 1. Аналитическая часть

В этой части рассматривается, что такое муравьиный алгоритм и его применение в решении задачи коммивояжёра.

## 1.1. Муравьиный алгоритм

Муравьиный алгоритм - один из эффективных [полиномиальных алгоритмов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B8%D0%BD%D0%BE%D0%BC%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) для нахождения приближённых решений [задачи коммивояжёра](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BC%D0%B8%D0%B2%D0%BE%D1%8F%D0%B6%D1%91%D1%80%D0%B0), а также решения аналогичных задач поиска маршрутов на [графах](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84_(%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)). Суть подхода заключается в анализе и использовании модели поведения [муравьёв](https://ru.wikipedia.org/wiki/%D0%9C%D1%83%D1%80%D0%B0%D0%B2%D0%B5%D0%B9), ищущих пути от колонии к источнику питания и представляет собой метаэвристическую оптимизацию.

В основе алгоритма лежит поведение [муравьиной колонии](https://ru.wikipedia.org/w/index.php?title=%D0%9C%D1%83%D1%80%D0%B0%D0%B2%D1%8C%D0%B8%D0%BD%D0%B0%D1%8F_%D0%BA%D0%BE%D0%BB%D0%BE%D0%BD%D0%B8%D1%8F&action=edit&redlink=1) — маркировка более удачных путей большим количеством [феромона](https://ru.wikipedia.org/wiki/%D0%A4%D0%B5%D1%80%D0%BE%D0%BC%D0%BE%D0%BD). Работа начинается с размещения муравьёв в вершинах графа (городах), затем начинается движение муравьёв — направление определяется вероятностным методом, на основании формулы вида (1), где n – число, обратно пропорциональное к расстоянию между городами I и J, t – количество феромона между этими городами.

(1)

В реальном мире [муравьи](https://ru.wikipedia.org/wiki/%D0%9C%D1%83%D1%80%D0%B0%D0%B2%D1%8C%D0%B8) (первоначально) ходят в случайном порядке и по нахождении продовольствия возвращаются в свою колонию, прокладывая [феромонами](https://ru.wikipedia.org/wiki/%D0%A4%D0%B5%D1%80%D0%BE%D0%BC%D0%BE%D0%BD) тропы. Если другие муравьи находят такие тропы, они, вероятнее всего, пойдут по ним. Вместо того, чтобы отслеживать цепочку, они укрепляют её при возвращении, если в конечном итоге находят источник питания. Со временем феромонная тропа начинает испаряться, тем самым уменьшая свою привлекательную силу. Чем больше времени требуется для прохождения пути до цели и обратно, тем сильнее испарится феромонная тропа. На коротком пути, для сравнения, прохождение будет более быстрым и как следствие, плотность феромонов остаётся высокой. Испарение феромонов также имеет свойство избежания стремления к локально-оптимальному решению. Если бы феромоны не испарялись, то путь, выбранный первым, был бы самым привлекательным. В этом случае, исследования пространственных решений были бы ограниченными. Таким образом, когда один муравей находит (например, короткий) путь от колонии до источника пищи, другие муравьи, скорее всего пойдут по этому пути, и положительные отзывы в конечном итоге приводят всех муравьёв к одному, кратчайшему, пути.

Правило обновления феромонов представлено формулой (2).

(2)

# 2. Конструкторская часть

В этой части будут составлен псевдокод алгоритмов.

## 2.1 Разработка алгоритмов

Псевдокод муравьиного алгоритма

procedure ACO\_MetaHeuristic

while(not\_termination)

generateSolutions()

daemonActions()

pheromoneUpdate()

end while

end procedure

# 3. Технологическая часть

В этой части обосновывается выбор языка программирования, необходимых библиотек и будет приведен листинг кодов каждого из алгоритмов.

## 3.1 Требования к программному обеспечению

Для того, чтобы запустить программу, необходимо иметь:

ОС windows 10, объем оперативной памяти не менее 512 мб, среда visual studio 2017.

## 3.2 Средства реализации

В качестве языка программирования был выбран C#. Его выбор основан на следующих причинах:

1. язык программирования C# является частью платформы .NET Framework, которая хорошо документирована, обладает хорошей стабильностью работы, и имеет богатый набор библиотечных решений;
2. C# позволяет программисту уделять больше времени написанию кода и не тратить время на побочные действия;
3. C# просто отлаживать, он не обладает уязвимостями языка C++, например, проблемами с указателями;
4. C# по скорости работы почти не уступает языку C++.

Время выполнения замеряется средством .NET под названием StopWatch;

## 3.3 Листинг кода

Листинг муравьиного алгоритма

class AcoProcessor

{

private Graph \_graph;

private AcoParameters \_acoParameters;

public struct AcoParameters

{

public double alpha, beta, q, tMax, p, pheromonMin, pheromonInitial;

}

public AcoProcessor(Graph graph)

{

\_graph = graph;

}

public double GetShortestPath(AcoParameters parameters, out int[] path)

{

\_acoParameters = parameters;

PheromonMatrix pheromonMatrix = new PheromonMatrix(\_graph.Size, parameters.pheromonInitial);

double shortest = Double.PositiveInfinity;

int[] shortestPath = null;

for (int i = 0; i < parameters.tMax; i++)

{

Ant ant = new Ant(0, \_graph, pheromonMatrix, parameters);

while(ant.CanMove())

ant.Move();

double antPathLen = ant.GetLength();

if (antPathLen < shortest)

{

shortest = antPathLen;

shortestPath = ant.Path;

}

}

path = shortestPath;

return shortest;

}

}

class Ant

{

private int \_currentCity;

private List<int> \_path = new List<int>();

private Graph \_graph;

private PheromonMatrix \_pheromonMatrix;

private AcoProcessor.AcoParameters \_parameters;

private Random \_rnd = new Random();

private bool \_isElite = false;

public Ant(int initialCity, Graph graph, PheromonMatrix pheromonMatrix, AcoProcessor.AcoParameters parameters)

{

\_currentCity = initialCity;

\_graph = graph;

\_pheromonMatrix = pheromonMatrix;

\_parameters = parameters;

}

public bool IsElite

{

get => \_isElite;

set => \_isElite = value;

}

private double GetProbability(int destCity, List<int> available)

{

double inversedDistance = \_parameters.q / \_graph[\_currentCity, destCity];

double pheromon = \_pheromonMatrix[\_currentCity, destCity];

inversedDistance = Math.Pow(inversedDistance, \_parameters.beta);

pheromon = Math.Pow(pheromon, \_parameters.alpha);

double summ = 0;

foreach (var city in available)

{

double id = \_parameters.q / \_graph[\_currentCity, city];

double ph = \_pheromonMatrix[\_currentCity, city];

id = Math.Pow(id, \_parameters.beta);

ph = Math.Pow(ph, \_parameters.alpha);

summ += id \* ph;

}

return inversedDistance \* pheromon / summ;

}

private List<KeyValuePair<int, double>> GetProbabilities(List<int> available)

{

List < KeyValuePair<int, double> > result = new List<KeyValuePair<int, double>>();

foreach (var city in available)

{

double prob = GetProbability(city, available);

KeyValuePair<int, double> pair = new KeyValuePair<int, double>(city, prob);

result.Add(pair);

}

return result;

}

private int ProbabilisticChoise(List<KeyValuePair<int, double>> citiesProbs)

{

double prevPoint = 0;

double rnd = \_rnd.NextDouble();

foreach (var cityProb in citiesProbs)

{

if (prevPoint <= rnd && rnd < prevPoint + cityProb.Value)

return cityProb.Key;

prevPoint += cityProb.Value;

}

throw new ArgumentException();

}

private int EliteChoise(List<KeyValuePair<int, double>> citiesProbs)

{

double max = citiesProbs[0].Value;

int maxIndex = 0;

foreach (var cityProb in citiesProbs)

{

if (cityProb.Value > max)

{

max = cityProb.Value;

maxIndex = cityProb.Key;

}

}

return maxIndex;

}

public int[] Path => \_path.ToArray();

public double GetLength()

{

double length = 0;

for (int i = 1; i < \_path.Count; i++)

{

int start = \_path[i - 1];

int end = \_path[i];

length += \_graph[start, end];

}

return length;

}

public void DoPheromon()

{

double length = GetLength();

double dt = \_parameters.q / length;

for (int i = 1; i < \_path.Count; i++)

{

int start = \_path[i - 1];

int end = \_path[i];

length += \_graph[start, end];

\_pheromonMatrix[start, end] \*= (1 - \_parameters.p);

\_pheromonMatrix[start, end] += dt;

if (\_pheromonMatrix[start, end] < \_parameters.pheromonMin)

\_pheromonMatrix[start, end] = \_parameters.pheromonMin;

}

}

public bool CanMove()

{

List<int> available = \_graph.GetConnections(\_currentCity);

available = Utils.Intersect(available, \_path);

return available.Count != 0;

}

public void Move()

{

List<int> available = \_graph.GetConnections(\_currentCity);

available = Utils.Intersect(available, \_path);

List<KeyValuePair<int, double>> citiesProbs = GetProbabilities(available);

int nexCity;

if (\_isElite)

{

nexCity = EliteChoise(citiesProbs);

}

else

{

nexCity = ProbabilisticChoise(citiesProbs);

}

\_path.Add(nexCity);

\_currentCity = nexCity;

}

}

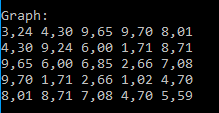
# 4. Экспериментальная часть

В этой части лабораторной работы будут поставлены эксперименты над реализованными алгоритмами и сделаны выводы.

## 4.1 Постановка эксперимента

Поставим эксперимент, в ходе которого подбреем такие исходные параметры алгоритма, при которых решение будет наилучшим.

Исходный граф:



Результаты перебора параметров:

alpha beta ro tMax Result

0,1 0,9 0,1 1 20,95

0,1 0,9 0,1 2 20,20

0,1 0,9 0,1 3 20,20

0,1 0,9 0,1 4 20,20

0,1 0,9 0,1 5 22,57

0,1 0,9 0,19 1 22,57

0,1 0,9 0,19 2 22,57

0,1 0,9 0,19 3 22,57

0,1 0,9 0,19 4 22,57

0,1 0,9 0,19 5 22,57

0,1 0,9 0,28 1 22,57

0,1 0,9 0,28 2 22,57

0,1 0,9 0,28 3 22,57

0,1 0,9 0,28 4 22,57

0,1 0,9 0,28 5 22,57

0,1 0,9 0,37 1 22,57

0,1 0,9 0,37 2 22,57

0,1 0,9 0,37 3 22,57

0,1 0,9 0,37 4 22,57

0,1 0,9 0,37 5 22,57

0,1 0,9 0,46 1 22,57

0,1 0,9 0,46 2 22,57

0,1 0,9 0,46 3 22,57

0,1 0,9 0,46 4 22,57

0,1 0,9 0,46 5 22,57

0,1 0,9 0,55 1 22,57

0,1 0,9 0,55 2 22,57

0,1 0,9 0,55 3 22,57

0,1 0,9 0,55 4 22,57

0,1 0,9 0,55 5 22,57

0,1 0,9 0,64 1 22,57

0,1 0,9 0,64 2 22,57

0,1 0,9 0,64 3 22,57

0,1 0,9 0,64 4 22,57

0,1 0,9 0,64 5 22,57

0,1 0,9 0,73 1 22,57

0,1 0,9 0,73 2 22,57

0,1 0,9 0,73 3 22,57

0,1 0,9 0,73 4 22,57

0,1 0,9 0,73 5 22,57

0,1 0,9 0,82 1 22,57

0,1 0,9 0,82 2 22,57

0,1 0,9 0,82 3 22,57

0,1 0,9 0,82 4 22,57

0,1 0,9 0,82 5 22,57

0,1 0,9 0,91 1 22,57

0,1 0,9 0,91 2 22,57

0,1 0,9 0,91 3 22,57

0,1 0,9 0,91 4 22,57

0,1 0,9 0,91 5 22,57

0,19 0,81 0,1 1 22,57

0,19 0,81 0,1 2 22,57

0,19 0,81 0,1 3 22,57

0,19 0,81 0,1 4 22,57

0,19 0,81 0,1 5 22,57

0,19 0,81 0,19 1 22,57

0,19 0,81 0,19 2 22,57

0,19 0,81 0,19 3 22,57

0,19 0,81 0,19 4 22,57

0,19 0,81 0,19 5 22,57

0,19 0,81 0,28 1 22,57

0,19 0,81 0,28 2 22,57

0,19 0,81 0,28 3 22,57

0,19 0,81 0,28 4 22,57

0,19 0,81 0,28 5 22,57

0,19 0,81 0,37 1 22,57

0,19 0,81 0,37 2 22,57

0,19 0,81 0,37 3 22,57

0,19 0,81 0,37 4 22,57

0,19 0,81 0,37 5 22,57

0,19 0,81 0,46 1 22,57

0,19 0,81 0,46 2 22,57

0,19 0,81 0,46 3 22,57

0,19 0,81 0,46 4 22,57

0,19 0,81 0,46 5 22,57

0,19 0,81 0,55 1 22,57

0,19 0,81 0,55 2 22,57

0,19 0,81 0,55 3 22,57

0,19 0,81 0,55 4 22,57

0,19 0,81 0,55 5 22,57

0,19 0,81 0,64 1 22,57

0,19 0,81 0,64 2 22,57

0,19 0,81 0,64 3 22,57

0,19 0,81 0,64 4 22,57

0,19 0,81 0,64 5 22,57

0,19 0,81 0,73 1 22,57

0,19 0,81 0,73 2 22,57

0,19 0,81 0,73 3 22,57

0,19 0,81 0,73 4 22,57

0,19 0,81 0,73 5 22,57

0,19 0,81 0,82 1 22,57

0,19 0,81 0,82 2 22,57

0,19 0,81 0,82 3 22,57

0,19 0,81 0,82 4 22,57

0,19 0,81 0,82 5 22,57

0,19 0,81 0,91 1 22,57

0,19 0,81 0,91 2 22,57

0,19 0,81 0,91 3 22,57

0,19 0,81 0,91 4 22,57

0,19 0,81 0,91 5 22,57

0,28 0,72 0,1 1 22,57

0,28 0,72 0,1 2 22,57

0,28 0,72 0,1 3 22,57

0,28 0,72 0,1 4 22,57

0,28 0,72 0,1 5 22,57

0,28 0,72 0,19 1 22,57

0,28 0,72 0,19 2 22,57

0,28 0,72 0,19 3 22,57

0,28 0,72 0,19 4 22,57

0,28 0,72 0,19 5 22,57

0,28 0,72 0,28 1 22,57

0,28 0,72 0,28 2 22,57

0,28 0,72 0,28 3 22,57

0,28 0,72 0,28 4 22,57

0,28 0,72 0,28 5 22,57

0,28 0,72 0,37 1 22,57

0,28 0,72 0,37 2 22,57

0,28 0,72 0,37 3 22,57

0,28 0,72 0,37 4 22,57

0,28 0,72 0,37 5 22,57

0,28 0,72 0,46 1 22,57

0,28 0,72 0,46 2 22,57

0,28 0,72 0,46 3 22,57

0,28 0,72 0,46 4 22,57

0,28 0,72 0,46 5 22,57

0,28 0,72 0,55 1 22,57

0,28 0,72 0,55 2 22,57

0,28 0,72 0,55 3 22,57

0,28 0,72 0,55 4 22,57

0,28 0,72 0,55 5 22,57

0,28 0,72 0,64 1 22,57

0,28 0,72 0,64 2 22,57

0,28 0,72 0,64 3 22,57

0,28 0,72 0,64 4 22,57

0,28 0,72 0,64 5 22,57

0,28 0,72 0,73 1 22,57

0,28 0,72 0,73 2 22,57

0,28 0,72 0,73 3 22,57

0,28 0,72 0,73 4 22,57

0,28 0,72 0,73 5 22,57

0,28 0,72 0,82 1 22,57

0,28 0,72 0,82 2 22,57

0,28 0,72 0,82 3 22,57

0,28 0,72 0,82 4 22,57

0,28 0,72 0,82 5 22,57

0,28 0,72 0,91 1 22,57

0,28 0,72 0,91 2 22,57

0,28 0,72 0,91 3 22,57

0,28 0,72 0,91 4 22,57

0,28 0,72 0,91 5 22,57

0,37 0,63 0,1 1 22,57

0,37 0,63 0,1 2 22,57

0,37 0,63 0,1 3 22,57

0,37 0,63 0,1 4 22,57

0,37 0,63 0,1 5 22,57

0,37 0,63 0,19 1 22,57

0,37 0,63 0,19 2 22,57

0,37 0,63 0,19 3 22,57

0,37 0,63 0,19 4 22,57

0,37 0,63 0,19 5 22,57

0,37 0,63 0,28 1 22,57

0,37 0,63 0,28 2 22,57

0,37 0,63 0,28 3 22,57

0,37 0,63 0,28 4 22,57

0,37 0,63 0,28 5 22,57

0,37 0,63 0,37 1 22,57

0,37 0,63 0,37 2 22,57

0,37 0,63 0,37 3 22,57

0,37 0,63 0,37 4 22,57

0,37 0,63 0,37 5 22,57

0,37 0,63 0,46 1 22,57

0,37 0,63 0,46 2 22,57

0,37 0,63 0,46 3 22,57

0,37 0,63 0,46 4 22,57

0,37 0,63 0,46 5 19,93

0,37 0,63 0,55 1 22,57

0,37 0,63 0,55 2 22,57

0,37 0,63 0,55 3 21,18

0,37 0,63 0,55 4 21,18

0,37 0,63 0,55 5 21,18

0,37 0,63 0,64 1 21,19

0,37 0,63 0,64 2 21,18

0,37 0,63 0,64 3 21,18

0,37 0,63 0,64 4 21,18

0,37 0,63 0,64 5 21,18

0,37 0,63 0,73 1 21,19

0,37 0,63 0,73 2 21,18

0,37 0,63 0,73 3 21,18

0,37 0,63 0,73 4 21,18

0,37 0,63 0,73 5 21,18

0,37 0,63 0,82 1 21,19

0,37 0,63 0,82 2 21,18

0,37 0,63 0,82 3 21,18

0,37 0,63 0,82 4 21,18

0,37 0,63 0,82 5 21,18

0,37 0,63 0,91 1 21,19

0,37 0,63 0,91 2 21,18

0,37 0,63 0,91 3 21,18

0,37 0,63 0,91 4 21,18

0,37 0,63 0,91 5 21,18

0,46 0,54 0,1 1 21,19

0,46 0,54 0,1 2 21,19

0,46 0,54 0,1 3 21,19

0,46 0,54 0,1 4 21,19

0,46 0,54 0,1 5 21,19

0,46 0,54 0,19 1 21,19

0,46 0,54 0,19 2 21,19

0,46 0,54 0,19 3 21,18

0,46 0,54 0,19 4 21,18

0,46 0,54 0,19 5 21,18

0,46 0,54 0,28 1 21,19

0,46 0,54 0,28 2 21,18

0,46 0,54 0,28 3 21,18

0,46 0,54 0,28 4 21,18

0,46 0,54 0,28 5 21,18

0,46 0,54 0,37 1 21,19

0,46 0,54 0,37 2 21,18

0,46 0,54 0,37 3 21,18

0,46 0,54 0,37 4 21,18

0,46 0,54 0,37 5 21,18

0,46 0,54 0,46 1 21,19

0,46 0,54 0,46 2 21,18

0,46 0,54 0,46 3 21,18

0,46 0,54 0,46 4 21,18

0,46 0,54 0,46 5 21,18

0,46 0,54 0,55 1 21,19

0,46 0,54 0,55 2 21,18

0,46 0,54 0,55 3 21,18

0,46 0,54 0,55 4 21,18

0,46 0,54 0,55 5 21,18

0,46 0,54 0,64 1 21,19

0,46 0,54 0,64 2 21,18

0,46 0,54 0,64 3 21,18

0,46 0,54 0,64 4 21,18

0,46 0,54 0,64 5 21,18

0,46 0,54 0,73 1 21,19

0,46 0,54 0,73 2 21,18

0,46 0,54 0,73 3 21,18

0,46 0,54 0,73 4 21,18

0,46 0,54 0,73 5 21,18

0,46 0,54 0,82 1 21,19

0,46 0,54 0,82 2 21,18

0,46 0,54 0,82 3 21,18

0,46 0,54 0,82 4 21,18

0,46 0,54 0,82 5 21,18

0,46 0,54 0,91 1 21,19

0,46 0,54 0,91 2 21,18

0,46 0,54 0,91 3 21,18

0,46 0,54 0,91 4 21,18

0,46 0,54 0,91 5 21,18

0,55 0,45 0,1 1 21,19

0,55 0,45 0,1 2 21,19

0,55 0,45 0,1 3 21,19

0,55 0,45 0,1 4 21,19

0,55 0,45 0,1 5 21,19

0,55 0,45 0,19 1 21,19

0,55 0,45 0,19 2 21,19

0,55 0,45 0,19 3 21,18

0,55 0,45 0,19 4 21,18

0,55 0,45 0,19 5 21,18

0,55 0,45 0,28 1 21,19

0,55 0,45 0,28 2 21,18

0,55 0,45 0,28 3 21,18

0,55 0,45 0,28 4 21,18

0,55 0,45 0,28 5 21,18

0,55 0,45 0,37 1 21,19

0,55 0,45 0,37 2 21,18

0,55 0,45 0,37 3 21,18

0,55 0,45 0,37 4 21,18

0,55 0,45 0,37 5 21,18

0,55 0,45 0,46 1 21,19

0,55 0,45 0,46 2 21,18

0,55 0,45 0,46 3 21,18

0,55 0,45 0,46 4 21,18

0,55 0,45 0,46 5 21,18

0,55 0,45 0,55 1 21,19

0,55 0,45 0,55 2 21,18

0,55 0,45 0,55 3 21,18

0,55 0,45 0,55 4 21,18

0,55 0,45 0,55 5 21,18

0,55 0,45 0,64 1 21,19

0,55 0,45 0,64 2 21,18

0,55 0,45 0,64 3 21,18

0,55 0,45 0,64 4 21,18

0,55 0,45 0,64 5 21,18

0,55 0,45 0,73 1 21,19

0,55 0,45 0,73 2 21,18

0,55 0,45 0,73 3 21,18

0,55 0,45 0,73 4 21,18

0,55 0,45 0,73 5 21,18

0,55 0,45 0,82 1 21,19

0,55 0,45 0,82 2 21,18

0,55 0,45 0,82 3 21,18

0,55 0,45 0,82 4 21,18

0,55 0,45 0,82 5 21,18

0,55 0,45 0,91 1 21,19

0,55 0,45 0,91 2 21,18

0,55 0,45 0,91 3 21,18

0,55 0,45 0,91 4 21,18

0,55 0,45 0,91 5 21,18

0,64 0,36 0,1 1 21,19

0,64 0,36 0,1 2 21,19

0,64 0,36 0,1 3 21,19

0,64 0,36 0,1 4 21,19

0,64 0,36 0,1 5 21,18

0,64 0,36 0,19 1 21,19

0,64 0,36 0,19 2 21,19

0,64 0,36 0,19 3 21,18

0,64 0,36 0,19 4 21,18

0,64 0,36 0,19 5 21,18

0,64 0,36 0,28 1 21,19

0,64 0,36 0,28 2 21,18

0,64 0,36 0,28 3 21,18

0,64 0,36 0,28 4 21,18

0,64 0,36 0,28 5 21,18

0,64 0,36 0,37 1 21,19

0,64 0,36 0,37 2 21,18

0,64 0,36 0,37 3 21,18

0,64 0,36 0,37 4 21,18

0,64 0,36 0,37 5 21,18

0,64 0,36 0,46 1 21,19

0,64 0,36 0,46 2 21,18

0,64 0,36 0,46 3 21,18

0,64 0,36 0,46 4 21,18

0,64 0,36 0,46 5 17,72

0,64 0,36 0,55 1 21,19

0,64 0,36 0,55 2 21,18

0,64 0,36 0,55 3 21,18

0,64 0,36 0,55 4 17,72

0,64 0,36 0,55 5 17,72

0,64 0,36 0,64 1 21,19

0,64 0,36 0,64 2 21,18

0,64 0,36 0,64 3 17,72

0,64 0,36 0,64 4 17,72

0,64 0,36 0,64 5 17,72

0,64 0,36 0,73 1 21,19

0,64 0,36 0,73 2 21,18

0,64 0,36 0,73 3 17,72

0,64 0,36 0,73 4 16,55

0,64 0,36 0,73 5 16,55

0,64 0,36 0,82 1 21,19

0,64 0,36 0,82 2 21,18

0,64 0,36 0,82 3 16,70

0,64 0,36 0,82 4 16,55

0,64 0,36 0,82 5 16,55

0,64 0,36 0,91 1 21,19

0,64 0,36 0,91 2 17,72

0,64 0,36 0,91 3 16,55

0,64 0,36 0,91 4 16,55

0,64 0,36 0,91 5 16,55

0,73 0,27 0,1 1 21,19

0,73 0,27 0,1 2 21,19

0,73 0,27 0,1 3 21,19

0,73 0,27 0,1 4 21,19

0,73 0,27 0,1 5 21,18

0,73 0,27 0,19 1 21,19

0,73 0,27 0,19 2 21,19

0,73 0,27 0,19 3 21,18

0,73 0,27 0,19 4 21,18

0,73 0,27 0,19 5 21,18

0,73 0,27 0,28 1 21,19

0,73 0,27 0,28 2 21,18

0,73 0,27 0,28 3 21,18

0,73 0,27 0,28 4 21,18

0,73 0,27 0,28 5 21,19

0,73 0,27 0,37 1 21,97

0,73 0,27 0,37 2 21,97

0,73 0,27 0,37 3 21,97

0,73 0,27 0,37 4 21,97

0,73 0,27 0,37 5 21,97

0,73 0,27 0,46 1 21,97

0,73 0,27 0,46 2 21,97

0,73 0,27 0,46 3 21,97

0,73 0,27 0,46 4 21,97

0,73 0,27 0,46 5 21,97

0,73 0,27 0,55 1 21,97

0,73 0,27 0,55 2 21,97

0,73 0,27 0,55 3 21,97

0,73 0,27 0,55 4 21,97

0,73 0,27 0,55 5 21,97

0,73 0,27 0,64 1 21,97

0,73 0,27 0,64 2 21,97

0,73 0,27 0,64 3 21,97

0,73 0,27 0,64 4 21,97

0,73 0,27 0,64 5 21,97

0,73 0,27 0,73 1 21,97

0,73 0,27 0,73 2 21,97

0,73 0,27 0,73 3 21,97

0,73 0,27 0,73 4 21,97

0,73 0,27 0,73 5 21,97

0,73 0,27 0,82 1 21,97

0,73 0,27 0,82 2 21,97

0,73 0,27 0,82 3 21,97

0,73 0,27 0,82 4 21,97

0,73 0,27 0,82 5 21,97

0,73 0,27 0,91 1 21,97

0,73 0,27 0,91 2 21,97

0,73 0,27 0,91 3 21,97

0,73 0,27 0,91 4 21,97

0,73 0,27 0,91 5 21,97

0,82 0,18 0,1 1 21,97

0,82 0,18 0,1 2 21,97

0,82 0,18 0,1 3 21,97

0,82 0,18 0,1 4 21,97

0,82 0,18 0,1 5 21,97

0,82 0,18 0,19 1 21,97

0,82 0,18 0,19 2 21,97

0,82 0,18 0,19 3 21,97

0,82 0,18 0,19 4 21,97

0,82 0,18 0,19 5 21,97

0,82 0,18 0,28 1 21,97

0,82 0,18 0,28 2 21,97

0,82 0,18 0,28 3 21,97

0,82 0,18 0,28 4 21,97

0,82 0,18 0,28 5 21,97

0,82 0,18 0,37 1 21,97

0,82 0,18 0,37 2 21,97

0,82 0,18 0,37 3 21,97

0,82 0,18 0,37 4 21,97

0,82 0,18 0,37 5 21,97

0,82 0,18 0,46 1 21,97

0,82 0,18 0,46 2 21,97

0,82 0,18 0,46 3 21,97

0,82 0,18 0,46 4 21,97

0,82 0,18 0,46 5 21,97

0,82 0,18 0,55 1 21,97

0,82 0,18 0,55 2 21,97

0,82 0,18 0,55 3 21,97

0,82 0,18 0,55 4 21,97

0,82 0,18 0,55 5 21,97

0,82 0,18 0,64 1 21,97

0,82 0,18 0,64 2 21,97

0,82 0,18 0,64 3 21,97

0,82 0,18 0,64 4 21,97

0,82 0,18 0,64 5 21,97

0,82 0,18 0,73 1 21,97

0,82 0,18 0,73 2 21,97

0,82 0,18 0,73 3 21,97

0,82 0,18 0,73 4 21,97

0,82 0,18 0,73 5 21,97

0,82 0,18 0,82 1 21,97

0,82 0,18 0,82 2 21,97

0,82 0,18 0,82 3 21,97

0,82 0,18 0,82 4 21,97

0,82 0,18 0,82 5 21,97

0,82 0,18 0,91 1 21,97

0,82 0,18 0,91 2 21,97

0,82 0,18 0,91 3 21,97

0,82 0,18 0,91 4 21,97

0,82 0,18 0,91 5 21,97

0,91 0.9 0,1 1 21,97

0,91 0.9 0,1 2 21,97

0,91 0.9 0,1 3 21,97

0,91 0.9 0,1 4 21,97

0,91 0.9 0,1 5 21,97

0,91 0.9 0,19 1 21,97

0,91 0.9 0,19 2 21,97

0,91 0.9 0,19 3 21,97

0,91 0.9 0,19 4 21,97

0,91 0.9 0,19 5 21,97

0,91 0.9 0,28 1 21,97

0,91 0.9 0,28 2 21,97

0,91 0.9 0,28 3 21,97

0,91 0.9 0,28 4 21,97

0,91 0.9 0,28 5 21,97

0,91 0.9 0,37 1 21,97

0,91 0.9 0,37 2 21,97

0,91 0.9 0,37 3 21,97

0,91 0.9 0,37 4 21,97

0,91 0.9 0,37 5 21,97

0,91 0.9 0,46 1 21,97

0,91 0.9 0,46 2 21,97

0,91 0.9 0,46 3 21,97

0,91 0.9 0,46 4 21,97

0,91 0.9 0,46 5 21,97

0,91 0.9 0,55 1 21,97

0,91 0.9 0,55 2 21,97

0,91 0.9 0,55 3 21,97

0,91 0.9 0,55 4 21,97

0,91 0.9 0,55 5 21,97

0,91 0.9 0,64 1 21,97

0,91 0.9 0,64 2 21,97

0,91 0.9 0,64 3 21,97

0,91 0.9 0,64 4 21,97

0,91 0.9 0,64 5 21,97

0,91 0.9 0,73 1 21,97

0,91 0.9 0,73 2 21,97

0,91 0.9 0,73 3 21,97

0,91 0.9 0,73 4 21,97

0,91 0.9 0,73 5 21,97

0,91 0.9 0,82 1 21,97

0,91 0.9 0,82 2 21,97

0,91 0.9 0,82 3 21,97

0,91 0.9 0,82 4 21,97

0,91 0.9 0,82 5 21,97

0,91 0.9 0,91 1 21,97

0,91 0.9 0,91 2 21,97

0,91 0.9 0,91 3 21,97

0,91 0.9 0,91 4 21,97

0,91 0.9 0,91 5 21,97

Отсюда видно, что наилучший результат для данного графа достигается при исходных параметрах

Степень при феромоне - 0,64

Степень при обратном расстоянии - 0,36

Скорость испарения - 0,82

Количество поколений – 4

# Заключение

Муравьиный алгоритм позволяет решить NP-полную задачу коммивояжёра за полиномиальное время, но требует изначальной настройки параметров.