

Государственное образовательное учреждение высшего профессионального образования

«Московский Государственный Технический Университет имени Н. Э. Баумана»

ОТЧЕТ

По лабораторной работе №7

По курсу «Анализ алгоритмов»

Тема: **«Муравьиный алгоритм»**

Студент: Кононенко С. Д.

Группа: ИУ7-51

Москва, 2017

Постановка задачи

1. Реализовать муравьиный алгоритм на языке программирования

2. Сравнить работу алгоритма при разных значениях параметров задающих веса феромона и времени жизни колонии

Идея

Муравьиный алгоритм - один из эффективных полиномиальных алгоритмов для нахождения приближённых решений задачи коммивояжёра, а также решения аналогичных задач поиска маршрутов на графах. Суть подхода заключается в анализе и использовании модели поведения муравьёв, ищущих пути от колонии к источнику питания и представляет собой метаэвристическую оптимизацию.

С полным разбором алгоритма можно ознакомиться в книге М.В.Ульянова «Ресурсно-эффективные компьютерные алгоритмы. Разработка и анализ» [7.4]

Реализация

static void gogo\_ant(ant\_t \*ant, matrix\_t adj\_mat, matrix\_t weight, matrix\_t d\_pheromon, int q)

{

int N = weight.n;

int i = 1;

int next = 0;

array\_t prob = create\_array(N);

while (i < N)

{

float sum\_weight = 0;

for (int j = 0; j < N; j++)

sum\_weight += weight.matr[ant->curr\_city][j] \* ant->Jk.arr[j];

for (int j = 0; j < N; j++)

{

prob.arr[j] = weight.matr[ant->curr\_city][j] / sum\_weight \* ant->Jk.arr[j];

}

next = choose\_next(prob);

ant->curr\_city = next;

ant->Jk.arr[next] = 0;

ant->route.arr[i++] = next;

}

ant->Lk = length\_of\_route(adj\_mat, ant->route);

add\_pheromon(d\_pheromon, ant->route, ant->Lk, q);

}

solution\_t solve(matrix\_t adj\_mat, float a, float b, float p, int q, int t\_max)

{

static int flag = 1;

if (flag)

{

srand(time(NULL));

flag = 0;

}

int N = adj\_mat.n;

ant\_t \*ants = create\_ant\_array(N); //array of ants, 1 per city;

matrix\_t pheromon = create\_matrix(N, N);

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

pheromon.matr[i][j] = 0.5;

matrix\_t visib = create\_matrix(N, N);

for (int i = 0; i < N; i++)

for (int j = i; j < N; j++)

if (i != j)

{

visib.matr[i][j] = 1 / adj\_mat.matr[i][j];

visib.matr[j][i] = visib.matr[i][j];

}

else

visib.matr[i][j] = 666;

matrix\_t weight = create\_matrix(N,N);

recalc\_weight(&weight, pheromon, visib, a, b);

matrix\_t d\_pheromon = create\_matrix(N, N);

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

d\_pheromon.matr[i][j] = 0;

int best\_l = INT\_MAX;

array\_t route = create\_array(N);

for (int t = 0; t < t\_max; t++)

{

for (int k = 0; k < N; k++)

{

init\_ant(&ants[k]);

gogo\_ant(&ants[k], adj\_mat, weight, pheromon, q);

}

int best = -1;

for (int i = 0; i < N; i++)

if (ants[i].Lk < best\_l)

{

best = i;

best\_l = ants[i].Lk;

}

if (best != -1)

copy\_array(route, ants[best].route);

recalc\_pheromon(&pheromon, &d\_pheromon, p);

recalc\_weight(&weight, pheromon, visib, a, b);

}

solution\_t solv = {best\_l, route};

free\_ant\_array(&ants, N);

free\_matrix(&pheromon);

free\_matrix(&weight);

free\_matrix(&visib);

free\_matrix(&d\_pheromon);

return solv;

}

***recalc\_weight()*** *и* ***recalc\_pheromon()*** *функции осуществляющие перерасчет матрицы весов и матрицы феромонов на ребрах после каждой итерации*

***choose\_next()*** *– фунция осуществляющая случайный выбор пути на основе массива вероятностей*

***length\_of\_route()*** *– функция считающая длину пройденного муравьем мути*

***add\_pheromon()*** *- функция изменяющая матрицу изменения кол-ва феромонов.*

Эксперимент

*Параметры эксперимента:*

*-кол-во итераций 200*

*-коэфициент испарения 0.5*

*-размерность матрицы смежностей 100\*100*

*-значения матрицы смежностей – случайные целые величины от 1 до 30*

|  |  |  |
| --- | --- | --- |
| α | β | Длина найденного маршрута |
| 0 | 1 | 534 |
| 0.1 | 0.9 | 534 |
| 0.2 | 0.8 | 449 |
| 0.3 | 0.7 | 401 |
| 0.4 | 0.6 | 391 |
| 0.5 | 0.5 | 388 |
| 0.6 | 0.4 | 372 |
| 0.7 | 0.3 | 371 |
| 0.8 | 0.2 | 392 |
| 0.9 | 0.1 | 569 |
| 1 | 0 | 1138 |

*Таблица 1. Резульаты эксперимента №1*

*Параметры эксперимента:*

*-коэффициент стадности(α) = 0.5*

*-коэффициент жадности(β) = 0.5*

*-коэфициент испарения 0.5*

*-размерность матрицы смежностей 100\*100*

*-значения матрицы смежностей – случайные целые величины от 1 до 30*

|  |  |
| --- | --- |
| Итерации | Длина найденного маршрута |
| 100 | 375 |
| 200 | 381 |
| 300 | 365 |
| 400 | 362 |
| 500 | 356 |
| 600 | 352 |
| 700 | 340 |
| 800 | 335 |
| 900 | 332 |
| 1000 | 321 |

*Таблица 2. Резульаты эксперимента №2*

**Вывод:** 1. Оптимальным является соотношение *α = 0.6±0.1, β = 0.4±0.1 , при при других значения, алгоритм зачастую находит путь далекий от оптимального.*

*2. Увеличение кол-ва итераций однозначно приводит к улучшению результата*

Заключение

В ходе лабораторной работы был реализован муравьиный алгоритм,а также было проведено сравнение работы алгоритма при различных параметрах, задающих веса феромона и временя жизни колонии.