Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Специальность: 1-40 81 02 «Технологии виртуализации и облачных вычислений»

ОТЧЕТ

по лабораторной работе

на тему

ПЧЕЛИНЫЙ АЛГОРИТМ ОПТИМИЗАЦИИ

по курсу

«Алгоритмы роевого поведения»

Выполнил: А.И. Андрадэ

Проверил: Д.Н. Одинец

МИНСК 2018

СОДЕРЖАНИЕ

[1 Цели лабораторной работы 4](#_Toc532154143)

[2 Общая постановка задачи на лабораторную работу 4](#_Toc532154144)

[3 Некоторые определения и допущения 4](#_Toc532154145)

[4 Муравьиный алгоритм оптимизации применительно к групповому поведению роботов. 5](#_Toc532154146)

[4.1 Блок-схемы алгоритма 6](#_Toc532154147)

[4.1 Шаги алгоритма 7](#_Toc532154148)

[5 Оценка качества выполнения групповой задачи по ресурсам 9](#_Toc532154149)

[5.1 Функционал качества E=f(N,t) выполнения групповой задачи от времени t 9](#_Toc532154150)

[5.2 Функционал качества E=f(N,t) выполнения групповой по ресурсам (количеству роботов на общую цель) 14](#_Toc532154151)

[ЗАКЛЮЧЕНИЕ 18](#_Toc532154152)

**1 Цели лабораторной работы**

1.1 Знакомство с алгоритмом роевого поведения.

1.2 Изучение и реализация алгоритма роевого поведения.

1.3 Моделирование *имитационной ситуации*.

**2 Общая постановка задачи на лабораторную работу**

Теоретическая часть представляет собой реферативный ответ на индивидуальный вопрос по одной из тем изучаемой дисциплины (согласно варианта задания).

Практическая часть представляет собой индивидуальную задачу по разработке законченного программного модуля на основе одной теоретической темы (один алгоритм роевого поведения).

В виду особенностей алгоритма, а также по договоренности с преподавателем, общая постановка задачи имеет следующий вид. При изучении целенаправленного поведения робота предлагается следующая оценка алгоритма поведения:

1) Размещение робота-агента на имитационном поле (карте местности);

2) Установка цели (точки назначения);

3) Оптимизация маршрута до цели.

**3 Некоторые определения и допущения**

Робот-агент–это муравейник (для муравьиного алгоритма оптимизации).

Роботы двигаются в двумерном евклидовом пространстве по ячейкам матрицы. Размер пространства по умолчанию – 40x40 ячеек.

Ближайшие ячейки, это ячейки, находящиеся непосредственно сверху, по бокам и снизу данной, их всего четыре. Расстояние между ближайшими ячейками – нормализовано и принято за 1.

Длина пути - кол-во ячеек, в которых побывал муравей.

Муравей имеет ориентацию в пространстве занимает одну ячейку и охотнее выбирает более близкую ячейку из окрестности, т.е. охотнее всего ту, которая находится прямо перед ним. Вероятности переходов в ячейки окрестности являются параметрами алгоритма и задают значения *эвристической информации* (см. дальше).

Имитационная ситуация – это комбинация робота-агента и цели (места доставки).

Феромонная карта местности – трехмерный массив размерностью 40x40x4, в котором i, j – ые значения – координаты ячейки, k –ое – массив значений феромонов для каждого направления.



Феромоны на границах карты принимают нулевые значения для того, чтобы муравей не выходил за границы.

Добавляются граничные условия на количество феромонов.

**4 Муравьиный алгоритм оптимизации применительно к групповому поведению роботов.**

Муравей - точка в двухмерном евклидовом пространстве, которое задается матрицей. В определенный момент времени муравей может находиться в ячейке матрицы. Ячейка определяет его позицию. У муравья также задана ориентация (значение из NSWE).

Цель муравьиной колонии (робота) – найти кратчайший путь между начальной позицией робота и позицией цели в пространстве (сетке).

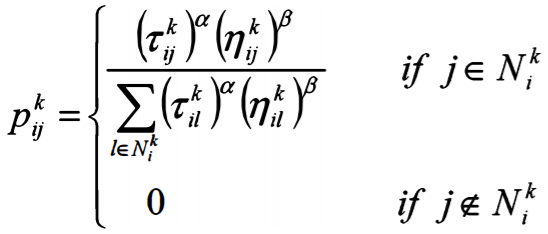
Во время поиска еды, муравьи показывают сложное социальное поведение, основанное на гормонах, которые они испускают. Феромоны привлекают других муравьев и отмечают путь к источнику пищи, которому

другие муравьи могут следовать. Чем больше муравьев идут по пути, тем больше феромонов выпускается, что увеличивает вероятность того, что больше муравьев пройдет по этому пути. Кратчайший путь к еде накапливает больше всего феромонов, потому что больше муравьев могут путешествовать за меньшее количество времени по нем.

Это явление впервые наблюдалось в знаменитом эксперименте с двойным мостом: когда предоставляется выбор между коротким путем и длинным путем к источнику пищи, муравьи постоянно находят кратчайший путь после

некоторого промежутка времени. Для предотвращения установления субоптимального пути (когда решение попадает в локальный минимум), феромон также испаряется со временем, таким образом уменьшая вроятность для других муравьев выбора данной дорожки. С другой стороны, уровни феромонов на кратчайшем пути остается высоким, потому что в этом случае скорость отложения феромона выше, чем скорость его испарения.

В пространстве (матрице) муравей может выбрать соседнюю ячейку с вероятностью, заданной формулой:



Где уровень феромона определяется , так же как и с реальными муравьями, чем больше феромона на пути, тем больше вероятность, что муравей последует этому пути. Сумма в знаменателе учитывает все возможные варианты выбора (или соседние ячейки, в реализиции алгоритма их четыре).

Парамерты α, β и  являются настраиваемыми параметрами алгоритма.

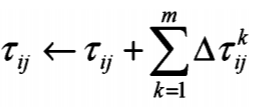
 - эвристическая информация, а α и β – вес (или важность) значений феромона и эвристической информации. Когда β = 0, , тогда вероятность только зависит от уровня феромона, с другой стороны, когда α = 0, вероятность зависит только от эвристической информации.

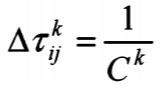
Значения феромона на пути (от узла i к j) может испаряться с процентом ρ (также называется скоростью испарения).



Где 0 ≤ ρ < 1.

После испарения феромона, к феромонной карте добавляются новые значения феромона тех муравьев, которые нашли новый путь и вернулись в муравейник.





Где  - назначенное значение награды k-ого муравья за выбор данного пути.

**4.1 Блок-схемы алгоритма**



**4.1 Шаги алгоритма**

Ядро алгоритма описано в файле Colony.py, класс – Colony, метод – find\_target().

def find\_target(self, target\_pos, proximity\_to\_standard=0.6, iter\_max=None):

best\_way = None

n\_iter = 0

way\_counter = 0

standard\_length = self.standard\_length(target\_pos)

satisfies\_accuracy = False

while True:

iter\_ways = []

*# испаряем феромоны*

*# их значения записаны в трехмерный массив*

*# x, y - координаты ячейки, z - массив значений веромонов, в зависимости от направлений*

self.pheromone\_map.pheromone\_evaporation()

for ant in self.ants:

*# каждый муравей дерает передвигается на следующую ячейку*

new\_pos = ant.motion.move()

*# если нашли цель, то сохраняем решение*

if new\_pos == target\_pos:

# то сохраняем решение

iter\_ways.append(ant.motion.path)

if best\_way is None:

best\_way = ant.motion.path

*# муравей возвращается в колонию, оставляя после себя феромонный след*

ant.motion.return\_to\_colony(len(best\_way))

way\_counter += 1

n\_iter += 1

*# если истекло макс кол-во итераций, вернуть лучшее на данный момент решение*

if iter\_max is not None and n\_iter >= iter\_max:

break

if not iter\_ways:

continue

pretendent = min(iter\_ways, key=len)

if best\_way is None or len(best\_way) > len(pretendent):

best\_way = pretendent

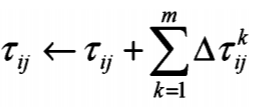
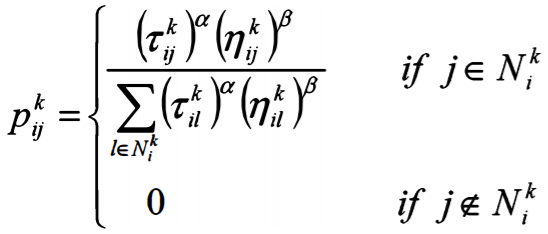
*# если достигнута достаточное подобие к эталону, вернуть решение*

if proximity\_to\_standard is not None and standard\_length / len(best\_way) >= proximity\_to\_standard:

satisfies\_accuracy = True

break

return best\_way, n\_iter, satisfies\_accuracy

* цикл
  + Испарение феромона (уменшаем все значения феромона по формуле . Значения феромона не могут быть меньше граничного.
  + Цикл для каждого муравья
    - Муравей выбирает и одну из соседних ячеек в соответствии с формулой . Параметр эвристики определяет то, что муравей c большей вероятностью продолжит движение по прямой, чем повернет или развернется.
    - Если муравей достиг цели, то сохраняем решение.
    - муравей возвращается в колонию, оставляя после себя феромонный след.
* Если истекло макс кол-во итераций, вернуть лучшее на данный момент решение.

**5 Оценка качества выполнения групповой задачи по ресурсам**

Задается N имитационных, в которых детерминированным является массив локальных целевых нагрузок, места доставки и количество роботов. Расстановка роботов на местности и назначение им локальных целевых нагрузок являются случайными процессами. Исследование целенаправленного группового поведения коллектива роботов выполняется на основе оценки времени доставки всей целевой нагрузки к месту назначения (времени выполнения групповой задачи) на протяжении N имитационных ситуаций.

Массив координат локальных целевых нагрузок loads=[(14, 33), (5, 35), (18, 23)].

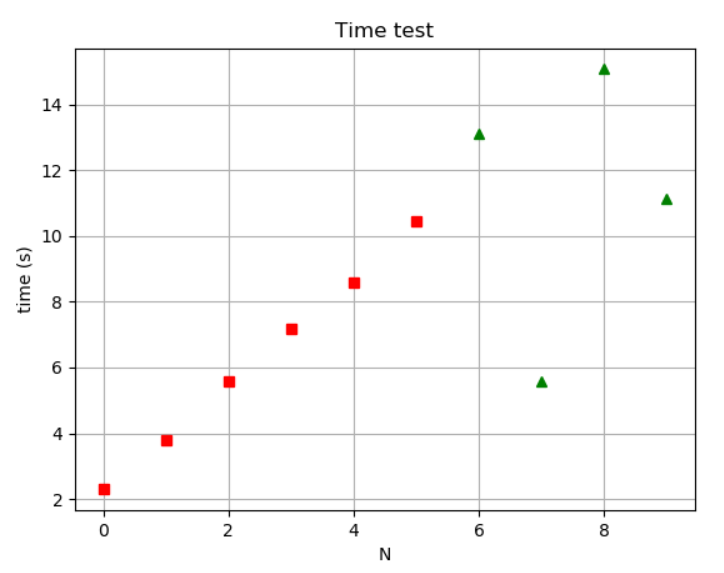
Массив координат мест доставки destinations=[(25, 25), (30, 30), (5, 18)].

**5.1 Функционал качества E=f(N,t) выполнения групповой задачи от времени t**

Размер колонии (colony\_size) фиксирован = 100 муравьев.

Кол-во итераций от 200 до 2000.

Кол-во имитационных ситуаций = 10.



Situation 0

robot: (25, 32) load: (14, 33) destination: (25, 25)

robot: (29, 19) load: (5, 35) destination: (30, 30)

robot: (29, 33) load: (18, 23) destination: (5, 18)

time = 2.2969086170196533

Situation 1

robot: (18, 26) load: (14, 33) destination: (25, 25)

robot: (4, 4) load: (5, 35) destination: (30, 30)

robot: (31, 2) load: (18, 23) destination: (5, 18)

time = 3.8036484718322754

Situation 2

robot: (27, 9) load: (14, 33) destination: (25, 25)

robot: (11, 23) load: (5, 35) destination: (30, 30)

robot: (2, 9) load: (18, 23) destination: (5, 18)

time = 5.584313631057739

Situation 3

robot: (35, 28) load: (14, 33) destination: (25, 25)

robot: (26, 0) load: (5, 35) destination: (30, 30)

robot: (0, 14) load: (18, 23) destination: (5, 18)

time = 7.164936304092407

Situation 4

robot: (37, 12) load: (14, 33) destination: (25, 25)

robot: (10, 15) load: (5, 35) destination: (30, 30)

robot: (14, 9) load: (18, 23) destination: (5, 18)

time = 8.585174322128296

Situation 5

robot: (27, 5) load: (14, 33) destination: (25, 25)

robot: (34, 12) load: (5, 35) destination: (30, 30)

robot: (12, 14) load: (18, 23) destination: (5, 18)

time = 10.434352397918701

Situation 6

robot: (25, 38) load: (14, 33) destination: (25, 25)

robot: (5, 2) load: (5, 35) destination: (30, 30)

robot: (2, 26) load: (18, 23) destination: (5, 18)

time = 13.105037450790405

Situation 7

robot: (4, 27) load: (14, 33) destination: (25, 25)

robot: (6, 2) load: (5, 35) destination: (30, 30)

robot: (11, 24) load: (18, 23) destination: (5, 18)

time = 5.584578514099121

Situation 8

robot: (35, 34) load: (14, 33) destination: (25, 25)

robot: (18, 4) load: (5, 35) destination: (30, 30)

robot: (10, 20) load: (18, 23) destination: (5, 18)

15.069006443023682

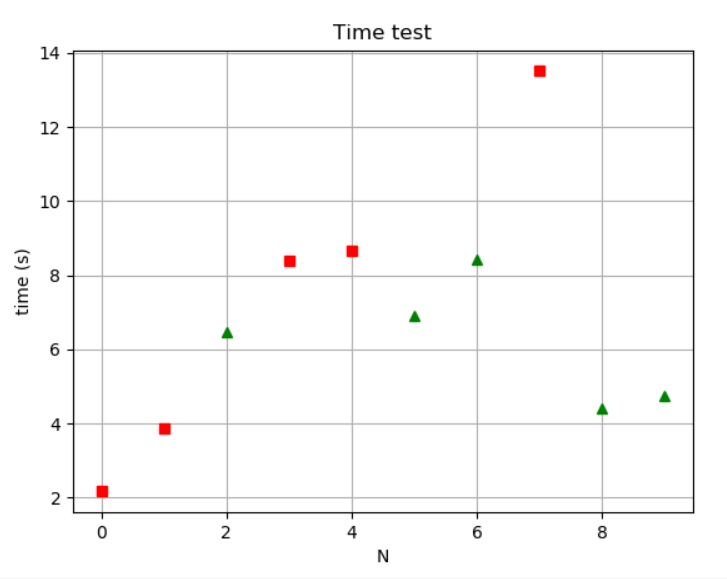
time = Situation 9

robot: (38, 2) load: (14, 33) destination: (25, 25)

robot: (2, 7) load: (5, 35) destination: (30, 30)

robot: (3, 33) load: (18, 23) destination: (5, 18)

time = 11.127280473709106



Situation 0

robot: (4, 31) load: (14, 33) destination: (25, 25)

robot: (27, 20) load: (5, 35) destination: (30, 30)

robot: (39, 34) load: (18, 23) destination: (5, 18)

time: 2.168843984603882

Situation 1

robot: (3, 12) load: (14, 33) destination: (25, 25)

robot: (23, 3) load: (5, 35) destination: (30, 30)

robot: (26, 5) load: (18, 23) destination: (5, 18)

time: 3.8588201999664307

Situation 2

robot: (2, 9) load: (14, 33) destination: (25, 25)

robot: (2, 16) load: (5, 35) destination: (30, 30)

robot: (21, 26) load: (18, 23) destination: (5, 18)

time: 6.457773447036743

Situation 3

robot: (22, 2) load: (14, 33) destination: (25, 25)

robot: (33, 29) load: (5, 35) destination: (30, 30)

robot: (38, 0) load: (18, 23) destination: (5, 18)

time: 8.40475869178772

Situation 4

robot: (35, 34) load: (14, 33) destination: (25, 25)

robot: (37, 23) load: (5, 35) destination: (30, 30)

robot: (5, 16) load: (18, 23) destination: (5, 18)

time: 8.676035404205322

Situation 5

robot: (20, 24) load: (14, 33) destination: (25, 25)

robot: (39, 12) load: (5, 35) destination: (30, 30)

robot: (25, 17) load: (18, 23) destination: (5, 18)

time: 6.905133485794067

Situation 6

robot: (4, 33) load: (14, 33) destination: (25, 25)

robot: (11, 22) load: (5, 35) destination: (30, 30)

robot: (29, 22) load: (18, 23) destination: (5, 18)

time: 8.42627239227295

Situation 7

robot: (37, 17) load: (14, 33) destination: (25, 25)

robot: (6, 17) load: (5, 35) destination: (30, 30)

robot: (6, 18) load: (18, 23) destination: (5, 18)

time: 13.509560108184814

Situation 8

robot: (38, 18) load: (14, 33) destination: (25, 25)

robot: (20, 33) load: (5, 35) destination: (30, 30)

robot: (3, 31) load: (18, 23) destination: (5, 18)

time: 4.405550956726074

Situation 9

robot: (6, 29) load: (14, 33) destination: (25, 25)

robot: (13, 24) load: (5, 35) destination: (30, 30)

robot: (29, 21) load: (18, 23) destination: (5, 18)

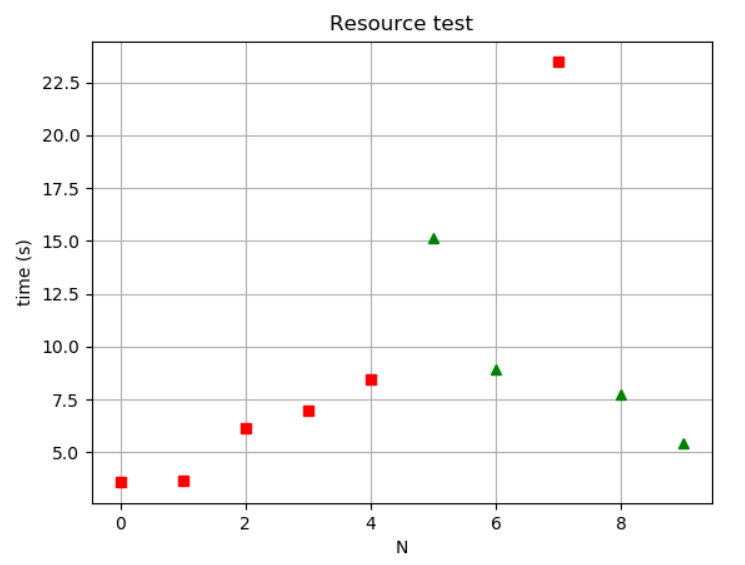
time: 4.7469096183776855

**5.2 Функционал качества E=f(N,t) выполнения групповой по ресурсам (количеству роботов на общую цель)**

Размер колонии (colony\_size) от 20 до 200 муравьев.

Кол-во итераций фиксирован, iter\_max = 1000.

Кол-во имитационных ситуаций = 10.



Situation 0

robot: (23, 22) load: (14, 33) destination: (25, 25)

robot: (27, 32) load: (5, 35) destination: (30, 30)

robot: (21, 1) load: (18, 23) destination: (5, 18)

time: 3.5826871395111084

Situation 1

robot: (9, 30) load: (14, 33) destination: (25, 25)

robot: (18, 13) load: (5, 35) destination: (30, 30)

robot: (17, 29) load: (18, 23) destination: (5, 18)

time: 3.641874074935913

Situation 2

robot: (37, 34) load: (14, 33) destination: (25, 25)

robot: (39, 22) load: (5, 35) destination: (30, 30)

robot: (5, 31) load: (18, 23) destination: (5, 18)

time: 6.165933847427368

Situation 3

robot: (38, 2) load: (14, 33) destination: (25, 25)

robot: (29, 24) load: (5, 35) destination: (30, 30)

robot: (20, 23) load: (18, 23) destination: (5, 18)

time: 6.95024561882019

Situation 4

robot: (39, 8) load: (14, 33) destination: (25, 25)

robot: (38, 0) load: (5, 35) destination: (30, 30)

robot: (22, 16) load: (18, 23) destination: (5, 18)

time: 8.428797245025635

Situation 5

robot: (11, 22) load: (14, 33) destination: (25, 25)

robot: (32, 0) load: (5, 35) destination: (30, 30)

robot: (33, 28) load: (18, 23) destination: (5, 18)

time: 15.122026205062866

Situation 6

robot: (20, 39) load: (14, 33) destination: (25, 25)

robot: (23, 14) load: (5, 35) destination: (30, 30)

robot: (4, 2) load: (18, 23) destination: (5, 18)

time: 8.91571831703186

Situation 7

robot: (21, 15) load: (14, 33) destination: (25, 25)

robot: (27, 20) load: (5, 35) destination: (30, 30)

robot: (7, 37) load: (18, 23) destination: (5, 18)

time: 23.463112831115723

Situation 8

robot: (2, 12) load: (14, 33) destination: (25, 25)

robot: (35, 21) load: (5, 35) destination: (30, 30)

robot: (29, 36) load: (18, 23) destination: (5, 18)

time: 7.745375394821167

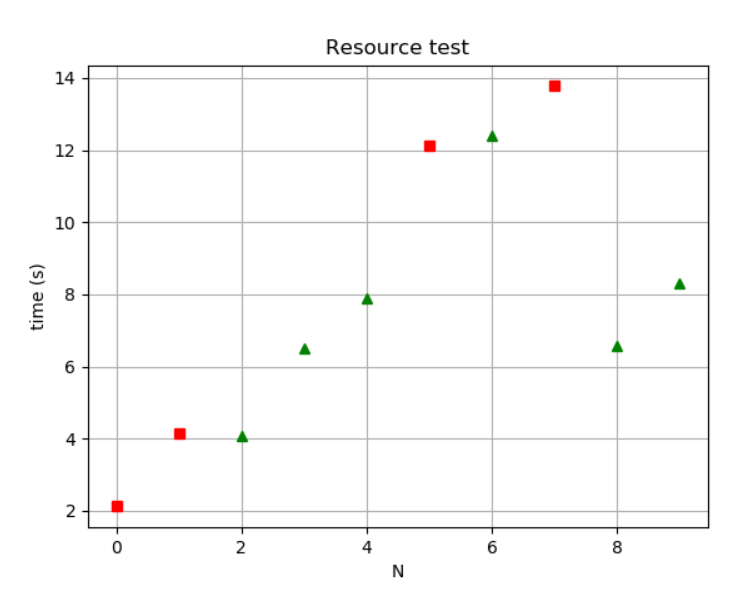
Situation 9

robot: (1, 15) load: (14, 33) destination: (25, 25)

robot: (22, 32) load: (5, 35) destination: (30, 30)

robot: (8, 19) load: (18, 23) destination: (5, 18)

time: 5.4323225021362305



Situation 0

robot: (5, 30) load: (14, 33) destination: (25, 25)

robot: (13, 26) load: (5, 35) destination: (30, 30)

robot: (33, 19) load: (18, 23) destination: (5, 18)

time: 2.1227521896362305

Situation 1

robot: (33, 11) load: (14, 33) destination: (25, 25)

robot: (10, 10) load: (5, 35) destination: (30, 30)

robot: (27, 31) load: (18, 23) destination: (5, 18)

time: 4.149887323379517

Situation 2

robot: (32, 39) load: (14, 33) destination: (25, 25)

robot: (17, 21) load: (5, 35) destination: (30, 30)

robot: (30, 23) load: (18, 23) destination: (5, 18)

time: 4.064460277557373

Situation 3

robot: (14, 34) load: (14, 33) destination: (25, 25)

robot: (13, 11) load: (5, 35) destination: (30, 30)

robot: (24, 5) load: (18, 23) destination: (5, 18)

time: 6.5144171714782715

Situation 4

robot: (5, 5) load: (14, 33) destination: (25, 25)

robot: (33, 39) load: (5, 35) destination: (30, 30)

robot: (8, 38) load: (18, 23) destination: (5, 18)

time: 7.895042657852173

Situation 5

robot: (21, 11) load: (14, 33) destination: (25, 25)

robot: (11, 13) load: (5, 35) destination: (30, 30)

robot: (8, 18) load: (18, 23) destination: (5, 18)

time: 12.1075730323791

Situation 6

robot: (36, 34) load: (14, 33) destination: (25, 25)

robot: (21, 4) load: (5, 35) destination: (30, 30)

robot: (32, 21) load: (18, 23) destination: (5, 18)

time: 12.407565593719482

Situation 7

robot: (35, 10) load: (14, 33) destination: (25, 25)

robot: (25, 34) load: (5, 35) destination: (30, 30)

robot: (33, 17) load: (18, 23) destination: (5, 18)

time: 13.77403998374939

Situation 8

robot: (26, 28) load: (14, 33) destination: (25, 25)

robot: (9, 10) load: (5, 35) destination: (30, 30)

robot: (30, 25) load: (18, 23) destination: (5, 18)

time: 6.5728490352630615

Situation 9

robot: (0, 38) load: (14, 33) destination: (25, 25)

robot: (23, 29) load: (5, 35) destination: (30, 30)

robot: (28, 27) load: (18, 23) destination: (5, 18)

time: 8.302802085876465

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы было выполнено ознакомление с теоретическими основами такого роевого алгоритма как пчелиный алгоритм оптимизации. Также была разработана программная реализация пчелиного алгоритма. Проведено моделирование имитационных ситуаций.

Таким образом, следует отметить, что цели, поставленные в лабораторной работе достигнуты, а сформулированные задачи выполнены в полном объеме.