## Лабораторная работа № 3

# Муравьиный алгоритм

**Цель работы.** Изучение и исследование параметров муравьиного алгоритма на примере решения задачи коммивояжера.

## ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Алгоритмы муравья (Ant algorithms), или оптимизация по принципу муравьиной колонии (это название было придумано изобретателем алгоритма, Марко Дориго (Marco Dorigo)), обладают специфическими свойствами, присущими муравьям, и используют их для ориентации в физическом пространстве. Алгоритмы муравья особенно интересны потому, что их можно использовать для решения не только статичных, но и динамических проблем, например, проблем маршрутизации в меняющихся сетях.

Хотя муравьи и слепы, они умеют перемещаться по сложной местности, находить пищу на большом расстоянии от муравейника и успешно возвращаться домой. Выделяя ферменты во время перемещения, муравьи изменяют окружающую среду, обеспечивают коммуникацию, а также отыскивают обратный путь в муравейник.

Самое удивительное в данном процессе - это то, что муравьи умеют находить самый оптимальный путь между муравейником и внешними точками. Чем больше муравьев используют один и тот же путь, тем выше концентрация ферментов на этом пути. Чем ближе внешняя точка к муравейнику, тем больше раз к ней перемещались муравьи. Что касается более удаленной точки, то ее муравьи достигают реже, поэтому по дороге к ней они применяют более сильные ферменты. Чем выше концентрация ферментов на пути, тем предпочтительнее он для муравьев по сравнению с другими доступными. Так муравьиная "логика" позволяет выбирать более короткий путь между конечными точками.

Алгоритмы муравья интересны, поскольку отражают ряд специфических свойств, присущих самим муравьям. Муравьи легко вступают в сотрудничество и работают вместе для достижения общей цели. Алгоритмы муравья работают так же, как муравьи. Это выражается в том, что смоделированные муравьи совместно решают проблему и помогают другим муравьям в дальнейшей оптимизации решения.

Рассмотрим пример. Два муравья из муравейника должны добраться до пищи, которая находится за препятствием. Во время перемещения каждый муравей выделяет немного фермента, используя его в качестве маркера.


Рис. 1. Начальная конфигурация (Tn).

При прочих равных условиях каждый муравей выберет свой путь. Первый муравей выбирает верхний путь, а второй - нижний. Так как нижний путь в два раза короче верхнего, второй муравей достигнет цели за время T1. Первый муравей в этот момент пройдет только половину пути (рис. 2).

Когда один муравей достигает пищи, он берет один из объектов и возвращается к муравейнику по тому же пути. За время T2 второй муравей вернулся в муравейник с пищей, а первый муравей достиг пищи (рис. 3).

Вспомните, что при перемещении каждого муравья на пути остается немного фермента. Для первого муравья за время T0-T2 путь был покрыт ферментом только один раз. В то же самое время второй муравей покрыл путь ферментом дважды. За время T4 первый муравей вернулся в муравейник, а второй муравей уже успел еще раз сходить к еде и вернуться. При этом концентрация фермента на нижнем пути будет в два раза выше, чем на верхнем. Поэтому первый муравей в следующий раз выберет нижний путь, поскольку там концентрация фермента выше.

В этом и состоит базовая идея алгоритма муравья - оптимизация путем непрямой связи между автономными агентами.

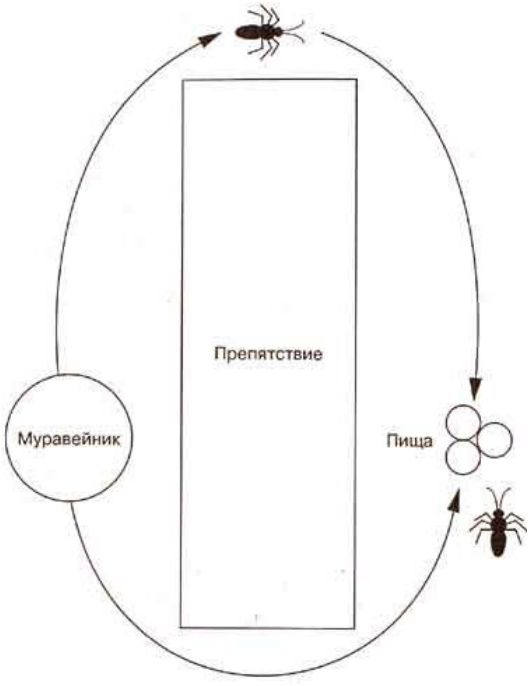



Рис. 2. Прошел один период времени (T1)

Алгоритм муравья

Подробно рассмотрим алгоритм муравья, чтобы понять, как он работает при решении конкретной проблемы.

Предположим, что окружающая среда для муравьев представляет собой закрытую двумерную сеть. Вспомните, что сеть - это группа узлов, соединенных посредством граней. Каждая грань имеет вес, который мы обозначим как расстояние между двумя узлами, соединенными ею. Граф двунаправленный, поэтому муравей может путешествовать по грани в любом направлении (рис. 4)

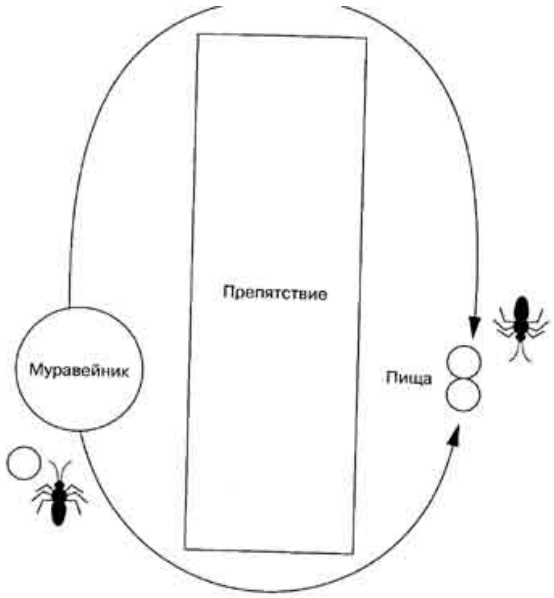
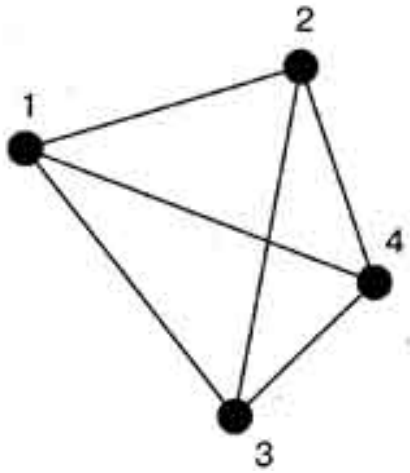



Рис. 3. Прошло два периода времени (T2)




**Граф с вершинами V= {1,2,3,4}**
**Грани E= {{1,2},{1,4},{1,3},{2,3},{2,4},{3,4}}**

Рис. 4. Пример полностью замкнутой двумерной сети V с набором граней E

Муравей - это программный агент, который является членом большой колонии и используется для решения какой-либо проблемы. Муравей снабжается набором простых правил, которые позволяют ему выбирать путь в

графе. Он поддерживает список табу (tabu list), то есть список узлов, которые он уже посетил. Таким образом, муравей должен проходить через каждый узел только один раз. Путь между двумя узлами графа, по которому муравей посетил каждый узел только один раз, называется путем Гамильтона (Hamiltonian path), по имени математика сэра Уильяма Гамильтона (Sir William Hamilton).

Узлы в списке "текущего путешествия" располагаются в том порядке, в котором муравей посещал их. Позже список используется для определения протяженности пути между узлами.

Настоящий муравей во время перемещения по пути будет оставлять за собой фермент. В алгоритме муравья агент оставляет фермент на гранях сети после завершения путешествия. О том, как это происходит, рассказывается в разделе "Путешествие муравья".

После создания популяция муравьев поровну распределяется по узлам сети. Необходимо равное разделение муравьев между узлами, чтобы все узлы имели одинаковые шансы стать отправной точкой. Если все муравьи начнут движение из одной точки, это будет означать, что данная точка является оптимальной для старта, а на самом деле мы этого не знаем.

*Движение муравья*

Движение муравья основывается на одном и очень простом вероятностном уравнении. Если муравей еще не закончил путь (path), то есть не посетил все узлы сети, для определения следующей грани пути используется уравнение 1.1:

$$P = (\mathrm{t}(r,u)^a \mathrm{h}(r,u)^b) / \sum_{\mathrm{k}} (\mathrm{t}(r,u)^a \mathrm{h}(r,u)^b) \qquad (1.1)$$

Здесь t(r,u) - интенсивность фермента на грани между узлами r и u, h(r,u) -функция, которая представляет измерение обратного расстояния для грани, a - вес фермента, а b коэффициент эвристики. Параметры a и b определяют относительную значимость двух параметров, а также их влияние на уравнение. Вспомните, что муравей путешествует только по узлам, которые еще не были посещены (как указано списком табу). Поэтому вероятность рассчитывается только для граней, которые ведут к еще не посещенным узлам. Переменная k представляет грани, которые еще не были посещены.

Пройденный муравьем путь отображается, когда муравей посетит все узлы диаграммы. Обратите внимание, что циклы запрещены, поскольку в алгоритм включен список табу. После завершения длина пути может быть подсчитана - она равна сумме всех граней, по которым путешествовал муравей. Уравнение 1.2 показывает количество фермента, который был оставлен на каждой грани пути для муравья k. Переменная Q является константой.

$$\mathrm{Dt}_{ij}^{k}(t) = Q/L^{k}(t) \qquad (1.2)$$

Результат уравнения является средством измерения пути, - короткий путь характеризуется высокой концентрацией фермента, а более длинный путь - более низкой. Затем полученный результат используется в уравнении 1.3, чтобы увеличить количество фермента вдоль каждой грани пройденного муравьем пути.

$$t_{ij}(t) = Dt_{ij}(t) + t^{k}_{ij}(t)r \qquad (1.3)$$

Обратите внимание, что данное уравнение применяется ко всему пути, при этом каждая грань помечается ферментом пропорционально длине пути. Поэтому следует дождаться, пока муравей закончит путешествие и только потом обновить уровни фермента, в противном случае истинная длина пути останется неизвестной. Константа r - значение между 0 и 1.

В начале пути у каждой грани есть шанс быть выбранной. Чтобы постепенно удалить грани, которые входят в худшие пути в сети, ко всем граням применяется процедура испарения фермента (Pheromone evaporation). Используя константу p из уравнения 1.3, мы получаем уравнение 1.4.

$$t_{ij}(t) = t_{ij}(t)(1-r) \qquad (1.4)$$

Поэтому для испарения фермента используется обратный коэффициент обновления пути.

После того как путь муравья завершен, грани обновлены в соответствии с длиной пути и произошло испарение фермента на всех гранях, алгоритм запускается повторно. Список табу очищается, и длина пути обнуляется. Муравьям разрешается перемещаться по сети, основывая выбор грани на уравнении 1.1

Этот процесс может выполняться для постоянного количества путей или до момента, когда на протяжении нескольких запусков не было отмечено повторных изменений. Затем определяется лучший путь, который и является решением.

**Пример итерации**

Давайте разберем функционирование алгоритма на простом примере, чтобы увидеть, как работают уравнения. Вспомните (рис. 1) простой сценарий с двумя муравьями, которые выбирают два разных пути для достижения одной цели. На рис. 5 показан этот пример с двумя гранями между двумя узлами (V0 и V1). Каждая грань инициализируется и имеет одинаковые шансы на то, чтобы быть выбранной.

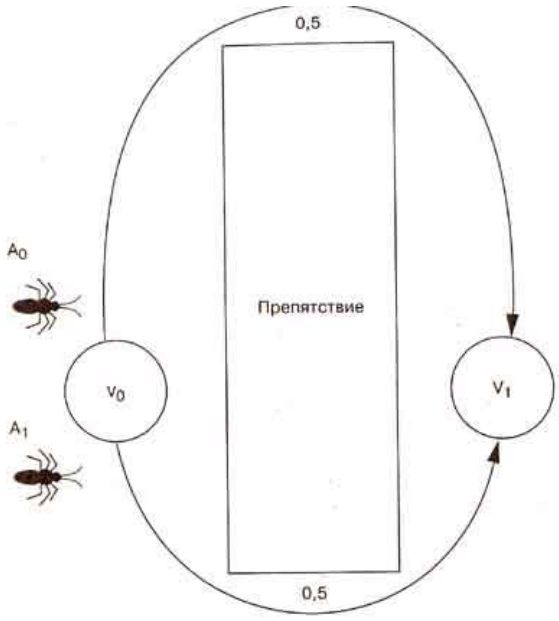



Рис. 5. Начальная конфигурация проблемы

Два муравья находятся в узле V0 помечаются как A0 и A1. Так как вероятность выбора любого пути одинакова, в этом цикле мы проигнорируем уравнение выбора пути. На рис. 4.6 каждый муравей выбирает свой путь (муравей A0 идет по верхнему пути, а муравей A1 - по нижнему).

В таблице на рис.6 показано, что муравей A0 сделал 20 шагов, а муравей A1, - только 10. По уравнению 2 мы рассчитываем количество фермента, которое должно быть "нанесено".

Примечание Работу алгоритма можно изменить, переопределив его параметры (например, r, a или b), например придав больший вес ферменту или расстоянию между узлами. Подробнее об этом рассказывается после обсуждения исходного кода.

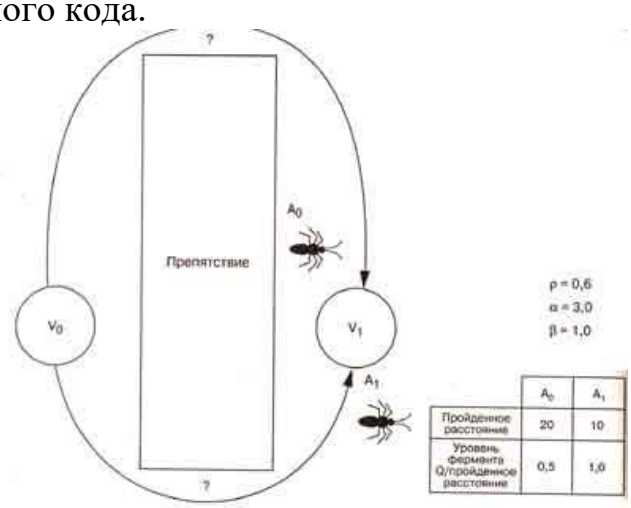



Рис. 6. Путь муравья завершен

Далее по уравнению 3 рассчитывается количество фермента, которое будет применено. Для муравья A0 результат составляет:

$$= 0,1 + 0,5 \cdot 0,6 = 0,4.$$

Для муравья A1 результат составляет: $0,1 + 1,0 \cdot 0,6 = 0,7$.

Далее с помощью уравнения 4 определяется, какая часть фермента испарится и, соответственно, сколько останется. Результаты (для каждого пути) составляют:

$$= 0,4 \cdot (1,0 - 0,6) = 0,16$$

$$- 0,7 \ (1,0 - 0,6) = 0,28.$$

Эти значения представляют новое количество фермента для каждого пути (верхнего и нижнего, соответственно). Теперь переместим Муравьев обратно в узел V0 воспользуемся вероятностным уравнением выбора пути 1.1, чтобы определить, какой путь должны выбрать муравьи.

Вероятность того, что муравей выберет верхний путь (представленный количеством фермента 0,16), составляет:

$$(0,16)^3 \cdot (0,5)^1 / ((0,16)^3 \cdot (0,5)^1 + ((0,28)^3 \cdot (1.0)^1) = 0,002048/0,024 = P(0,085).$$

Вероятность того, что муравей выберет нижний путь (0.28 фермента) составляет:

$$(0,28)^3 \cdot (1,0)^1 / ((0,16)^3 \cdot (0.5)^1) + ((0,28)^3 \cdot (1,0)^1) = 0,021952 / 0,024 = P(0,915).$$

При сопоставлении двух вероятностей оба муравья выберут нижний путь, который является наиболее оптимальным.

## ПРАКТИЧЕСКАЯ ЧАСТЬ

1. Реализовать windows-приложение, позволяющее решить задачу коммивояжера для графа с количеством вершин не менее 20, с использованием муравьиного алгоритма. Параметры алгоритма задаются пользователем.
2. Интерфейс приложения должен обеспечивать визуализацию процесса поиска кратчайшего пути, а также ввод параметров алгоритма.
3. Исследовать алгоритм на сходимость. При каких значения параметров или их комбинациях, алгоритм сходится наилучшим образом. Итоги и параметры экспериментов зафиксировать в табличном виде.

## СОДЕРЖАНИЕ ОТЧЕТА

1. Наименование и цель работы.
2. Блок-схема алгоритма решения задачи комивояжера с использованием муравьиного алгоритма.
3. Исходный код модулей программного обеспечения.
4. Результаты исследования влияния параметров алгоритма на его эффективность, оформленные в виде таблицы.
5. Выводы по работе.

## ВОПРОСЫ К ЗАЩИТЕ

1. Для решения каких проблем выгодно использовать алгоритмы муравья?
2. Какой ряд специфических свойств отражает алгоритм муравья?
3. Если граф двунаправленный, то «путешествие» может быть продолжено с места остановки в каком направлении?
4. Что координально отличает смысл алгоритма муравья и теории графов?