Проект

Использование муравьиных алгоритмов для решения задачи коммивояжера

С. Березовская githubmark.png

2021 год

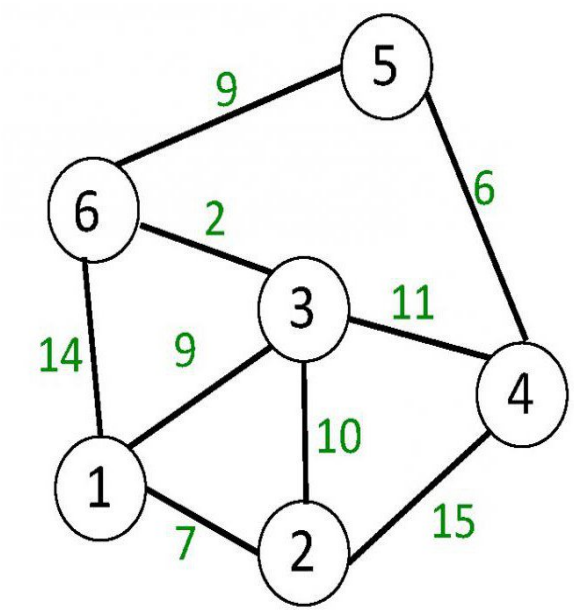
Содержание

1. Введение

Задача формулируется как задача поиска минимального по стоимости замкнутого маршрута по всем вершинам без повторений на полном взвешенном графе с n вершинами. Содержательно вершины графа являются городами, которые должен посетить коммивояжёр, а веса рёбер отражают расстояния (длины) или стоимости проезда. Эта задача является NP-трудной, и точный переборный алгоритм её решения имеет факториальную сложность.

```
     #include <iostream>
 #include <stdio.h>
    #define inf 1E9
    #define NMAX 16
    #define min(x, y) x < y ? x : y
     using namespace std;
    int n,i,j,k,m,temp,ans,d[NMAX][NMAX],t[1<<NMAX][NMAX];
    int get(int nmb,int x)
    { return (x&(1<<nmb))!=0; }
    int main(void)
    {
    cin>>n;
    for (i=0;i<n;++i)
    for (j=0;j<n;++j) cin >>d[i][j];
    t[1][0]=0; m=1<<n;
    for (i=1;i<m;i+=2)
    for (j=(i==1)?1:0;j<n;++j)
    {
    t[i][j]=inf;
    if (j>0 && get(j,i))
    {
    temp=i^(1<<j);
    for (k=0;k<n;++k)
    if (get(k,i) && d[k][j]>0) t[i][j]=min(t[i][j],t[temp][k]+d[k][j]);
    }
    }
    for (j=1,ans=inf;j<n;++j)
    if (d[j][0]>0) ans=min(ans,t[m-1][j]+d[j][0]);
    if (ans==inf) printf("-1"); else printf("%d", ans);
    return 0;
    }
```

Жадный алгоритм в данном случае может не сработать. Например



Если мы пойдем в этом графе от города 1 то выберем путь 1-2-3-6-5-4. Он не является наименьшим и из 4 мы не можем попасть в 1.

2. Биологические принципы поведения муравьиной колониии

Муравьи относятся к социальным насекомым, образующим коллективы. Коллективная система способна решать сложные динамические задачи по выполнению совместной работы, которая не могла бы выполняться каждым элементом системы в отдельности в разнообразных средах без внешнего управления, контроля или координации. В таких случаях говорят о роевом интеллекте (Swarm intelligence), как о замысловатых способах кооперативного поведения, то есть стратегии выживания.

Одним из подтверждений оптимальности поведения муравьиных колоний является тот факт, что сеть гнёзд суперколоний близка к минимальному остовному дереву графа их муравейников.

Основу поведения муравьиной колонии составляет самоорганизация, обеспечивающая достижения общих целей колонии на основе низкоуровневого взаимодействия. Колония не имеет централизованного управления, и её особенностями являются обмен локальной информацией только между отдельными особями (прямой обмен - пища, визуальные и химические контакты) и наличие непрямого обмена, который и используется в муравьиных алгоритмах. Таким образом, в общем случае рассматриваются слепые муравьи, не способные чувствовать близость пищи.

Непрямой обмен - стигмержи (stigmergy), представляет собой разнесённое во времени взаимодействие, при котором одна особь изменяет некоторую область окружающей среды, а другие используют эту информацию позже, когда в неё попадают. Биологи установили, что такое отложенное взаимодействие происходит через специальное химическое вещество - феромон (pheromone), секрет специальных желёз, откладываемый при перемещении муравья. Концентрация феромона на пути определяет предпочтительность движения по нему.

Адаптивность поведения реализуется испарением феромона, который в природе воспринимается муравьями в течение нескольких суток. Мы можем провести некоторую аналогию между распределением феромона в окружающем колонию пространстве, и «глобальной» памятью муравейника, носящей динамический характер.

3. Муравьиные алгоритмы

Муравьиные алгоритмы представляют собой вероятностную жадную эвристику, где вероятности устанавливаются, исходя из информации о качестве решения, полученной из предыдущих решений. Они могут использоваться как для статических, так и для динамических комбинаторных оптимизационных задач. Сходимость гарантирована, то есть в любом случае мы получим оптимальное решение, однако скорость сходимости неизвестна.

3.1. Немного из истории создания муравьиных алгоритмов

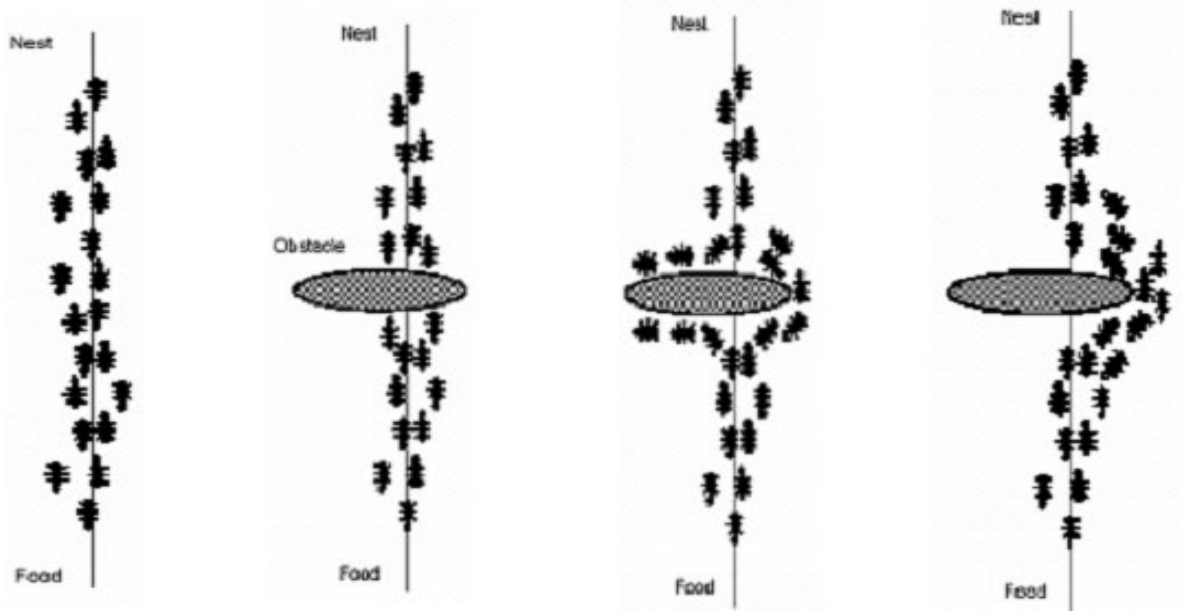
Началось всё с изучения поведения реальных муравьёв. Эксперименты с Argentine ants, проводимые Госсом в 1989 и Денеборгом в 1990 году послужили отправной точкой для дальнейшего исследования роевого интеллекта. Исследования применения полученных знаний для дискретной математики начались в начале 90-х годов XX века, автором идеи является Марко Дориго из Университета Брюсселя, Бельгия. Именно он впервые сумел формализовать поведение муравьёв и применить стратегию их поведения для решения задачи о кратчайших путях. Позже при участии Гамбарделлы, Тайлларда и Ди Каро были разработаны и многие другие подходы к решению сложных оптимизационных задач при помощью муравьиных алгоритмов. На сегодняшний день эти методы являются весьма конкурентоспособными по сравнению с другими эвристиками и для некоторых задач дают наилучшие на сегодняшний день результаты.

3.2. Концепция муравьиных алгоритмов

Идея муравьиного алгоритма - моделирование поведения муравьёв, связанного с их способностью быстро находить кратчайший путь от муравейника к источнику пищи и адаптироваться к изменяющимся условиям, находя новый кратчайший путь. При своём движении муравей метит путь феромоном, и эта информация используется другими муравьями для выбора пути. Это элементарное правило поведения и определяет способность муравьёв находить новый путь, если старый оказывается недоступным.

Рассмотрим случай, показанный на рисунке, когда на оптимальном доселе пути возникает преграда. В этом случае необходимо определение нового оптимального пути. Дойдя до преграды, муравьи с равной вероятностью будут обходить её справа и слева. То же самое будет происходить и на обратной стороне преграды. Однако, те муравьи, которые случайно выберут кратчайший путь, будут быстрее его проходить, и за несколько передвижений он будет более обогащён феромоном. Поскольку движение муравьёв определяется концентрацией феромона, то следу-

ющие будут предпочитать именно этот путь, продолжая обогащать его феромоном до тех пор, пока этот путь по какой-либо причине не станет недоступен.



Очевидная положительная обратная связь быстро приведёт к тому, что кратчайший путь станет единственным маршрутом движения большинства муравьёв. Моделирование испарения феромона - отрицательной обратной связи - гарантирует нам, что найденное локально оптимальное решение не будет единственным - муравьи будут искать и другие пути. Если мы моделируем процесс такого поведения на некотором графе, рёбра которого представляют собой возможные пути перемещения муравьёв, в течение определённого времени, то наиболее обогащённый феромоном путь по рёбрам этого графа и будет являться решением задачи, полученным с помощью муравьиного алгоритма.

3.3. Обобщённый алгоритм

Любой муравьиный алгоритм, независимо от модификаций, представим в следующем виде •Пока (условия выхода не выполнены)

1. Создаём муравьёв
2. Ищем решения
3. Обновляем феромон
4. Дополнительные действия опционально

Теперь рассмотрим каждый шаг в цикле более подробно

1. Создаём муравьёв

   • Стартовая точка, куда помещается муравей, зависит от ограничений, накладываемых условиями задачи. Потому что для каждой задачи способ размещение муравьёв является определяющим. Либо все они помещаются в одну точку, либо в разные с повторениями, либо без повторений.

• На этом же этапе задаётся начальный уровень феромона. Он инициализируется небольшим положительным числом для того, чтобы на начальном шаге вероятности перехода в следующую вершину не были нулевыми.

2. Ищем решения

• Вероятность перехода из вершины i в вершину j определяется по следующей формуле

$$P_{ij}(t) = \frac{\tau_{ij}(t)^{\alpha}(\frac{1}{d_{ij}})^{\beta}}{\sum_{i=1}^{nodes} \tau_{ij}(t)^{\alpha}(\frac{1}{d_{ij}})^{\beta}} \quad (1)$$

Где $\tau_{ij}(t)$- уровень феромона, d - эвристическое расстояние, $\alpha, \beta - .\alpha = 0, .$ При $\beta = 0, ., .$

```
int GetRandomNumber(){
int num = 1+ rand() % 1000;
return num;
}

int main(){
srand(time(0));
double  P, lam, d, al, bet;
double *p= new double [4];
for (int i=0;i<4; i++){
cin>>lam >> al;//3xdfg>>d >> bet;
p[i]=pow(lam,al);//*pow(1/d,bet);
P+=p[i];
}
cout<< endl;
int s=0;//составим все числа на отрезке от одного до 1000
for (int i=0;i<4; i++) {
p[i]/=P;
p[i]=p[i]*1000;
p[i]=(int)p[i]+s;
s=p[i];
cout<< p[i]<<" ";
}
int n;
n=GetRandomNumber();
cout << n << " random nomber " << endl;
if (n<=p[0]) cout<<0;
for (int i=1;i<4; i++) {
if ((n>p[i-1])&&(n<=p[i])) cout<<i;
```

```
}
```

3. Обновляем феромон • Уровень феромона обновляется в соответствии с приведённой формулой

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{i=1}^{usingway} \frac{Q}{L_k} \tag{2}$$

Где ρ - интенсивность испарения, L(t) - цена текущего решения для k-ого муравья, а Q - параметр, имеющий значение порядка цены оптимального решения, то есть $\frac{Q}{L_k(t)}$ - феромон, откладываемый k -ым муравьём, использующим ребро (i,j).

4. Дополнительные действия

• Обычно здесь используется алгоритм локального поиска, однако он может также появиться и после поиска всех решений.

3.4. Этапы решения задачи при помощи муравьиных алгоритмов

Для того чтобы построить подходящий муравьиный алгоритм для решения какой-либо задачи, нужно

1. Представить задачу в виде набора компонент и переходов или набором неориентированных взвешенных графов, на которых муравьи могут строить решения
2. Определить значение следа феромона
3. Определить эвристику поведения муравья, когда строим решение
4. Если возможно, то реализовать эффективный локальный поиск
5. Выбрать специфический ACO алгоритм и применить для решения задачи
6. Настроить параметр ACO алгоритма

Также определяющими являются

- Количество муравьёв
- Баланс между изучением и использованием
- Сочетание с жадными эвристиками или локальным поиском
- Момент, когда обновляется феромон

4. Обзор модификаций классического алгоритма

4.1. Концепция муравьиных алгоритмов

Результаты первых экспериментов с применением муравьиного алгоритма для решения задачи коммивояжера были многообещающими, однако далеко не лучшими по сравнению с уже существовавшими методами. Однако простота классического муравьиного алгоритма (названного «муравьиной системой») оставляла возможности для доработок – и именно алгоритмические усовершенствования стали предметом дальнейших исследований Марко Дориго и других специалистов в области комбинаторной оптимизации. В основном, эти усовершенствования связаны с большим использованием истории поиска и более тщательным исследованием областей вокруг уже найденных удачных решений. Ниже рассмотрены наиболее примечательные из модификаций. Сложности возникают с тем, что из последнего города, в который мы войдем может не быть пути в исходный. Также вероятностный алгоритм дает шанс для перехода по "плохому"пути и, если нам не везет несколько раз в начале, то весь алгоритм "ломается".

4.2. Elitist Ant System

Одним из таких усовершенствований является введение в алгоритм так называемых «элитных муравьев». Опыт показывает, что проходя ребра, входящие в короткие пути, муравьи с большей вероятностью будут находить еще более короткие пути. Таким образом, эффективной стратегией является искусственное увеличение уровня феромонов на самых удачных маршрутах. Для этого на каждой итерации алгоритма каждый из элитных муравьев проходит путь, являющийся самым коротким из найденных на данный момент.

Эксперименты показывают, что, до определенного уровня, увеличение числа элитных муравьев является достаточно эффективным, позволяя значительно сократить число итераций алгоритма. Однако, если число элитных муравьев слишком велико, то алгоритм достаточно быстро находит субоптимальные решение и застревает в нем. Как и другие изменяемые параметры, оптимальное число элитных муравьев следует определять опытным путем.

4.3. Ant-Q

Лука Гамбарделла (Luca M. Gambardella) и Марко Дориго опубликовали в 1995 году работу, в которой они представили муравьиный алгоритм, получивший свое название по аналогии с методом машинного обучения Q-learning . В основе алго-

ритма лежит идея о том, что муравьиную систему можно интерпретировать как систему обучения с подкреплением. Ant-Q усиливает эту аналогию, заимствуя многие идеи из Q-обучения.

Алгоритм хранит Q-таблицу, сопоставляющую каждому из ребер величину, определяющую «полезность» перехода по этому ребру. Эта таблица изменяется в процессе работы алгоритма – то есть обучения системы. Значение полезности перехода по ребру вычисляется исходя из значений полезностей перехода по следующим ребрам в результате предварительного определения возможных следующих состояний. После каждой итерации полезности обновляются исходя из длин путей, в состав которых были включены соответствующие ребра.

4.4. Ant Colony System

В 1997 году те же исследователи опубликовали работу, посвященную еще одному разработанному ими муравьиному алгоритму . Для повышения эффективности по сравнению с классическим алгоритмом, ими были введены три основных изменения.

Во-первых, уровень феромонов на ребрах обновляется не только в конце очередной итерации, но и при каждом переходе муравьев из узла в узел. Во-вторых, в конце итерации уровень феромонов повышается только на кратчайшем из найденных путей. В-третьих, алгоритм использует измененное правило перехода: либо, с определенной долей вероятности, муравей безусловно выбирает лучшее – в соответствие с длиной и уровнем феромонов – ребро, либо производит выбор так же, как и в классическом алгоритме.

4.5. Max-min Ant System

В том же году Томас Штютцле (Tomas Stützle) и Хольгер Хоос (Holger Hoos) предложили муравьиный алгоритм, в котором повышение концентрации феромонов происходит только на лучших путях из пройденных муравьями . Такое большое внимание к локальным оптимумам компенсируется вводом ограничений на максимальную и минимальную концентрацию феромонов на ребрах, которые крайне эффективно защищают алгоритм от преждевременной сходимости к субоптимальным решениям.

На этапе инициализации, концентрация феромонов на всех ребрах устанавливается равной максимальной. После каждой итерации алгоритма только один муравей оставляет за собой след – либо наиболее успешный на данной итерации, либо, аналогично алгоритму с элитизмом, элитный. Этим достигается, с одной стороны, более тщательное исследование области поиска, с другой – его ускорение.

4.6. ASrank

Бернд Бульнхаймер (Bernd Bullnheimer), Рихард Хартл (Richard F. Hartl) и Кристине Штраусс (Christine Strauß) разработали модификацию классического муравьиного алгоритма, в котором в конце каждой итерации муравьи ранжируются в соответствие с длинами пройденных ими путей . Количество феромонов, оставляемого муравьем на ребрах, таким образом, назначается пропорционально его позиции. Кроме того, для более тщательного исследования окрестностей уже найденных удачных решений, алгоритм использует элитных муравьев.

5. Алгоритм Литтла

1. В каждой строке матрицы стоимости найдем минимальный элемент и вычтем его из всех элементов строки. Сделаем это и для столбцов, не содержащих нуля. Получим матрицу стоимости, каждая строка и каждый столбец которой содержат хотя бы один нулевой элемент.
2. Для каждого нулевого элемента матрицы cij рассчитаем коэффициент Гi,j, который равен сумме наименьшего элемента i строки (исключая элемент Ci,j=0) и наименьшего элемента j столбца. Из всех коэффициентов Гi,j выберем такой, который является максимальным Гk,l=maxГi,j. В гамильтонов контур вносится соответствующая дуга (k,l).
3. Удаляем k-тую строку и столбец l, поменяем на бесконечность значение элемента Cl,k (поскольку дуга (k,l) включена в контур, то обратный путь из l в k недопустим). Повторяем алгоритм шага 1, пока порядок матрицы не станет равным двум. Затем в текущий ориентированный граф вносим две недостающие дуги, определяющиеся однозначно матрицей прядка 2. Получаем гамильтонов контур.

В ходе решения ведется постоянный подсчет текущего значения нижней границы. Нижняя граница равна сумме всех вычтенных элементов в строках и столбцах. Итоговое значение нижней границы должно совпасть с длиной результирующего контура. (Его можно использовать для поиска локального минимума)

6. Использованные источники

1. habr.com
2. wikipedia.org