Санкт-Петербургский политехнический университет
Высшая школа прикладной математики и вычислительной физики, ФизМех

Направление подготовки
«01.03.02 Прикладная математика и информатика»
Специальность «Системное программирование»

Курсовая работа
тема "Сравнительный анализ реализации задачи коммивояжёра с усложнением"
дисциплина "Методы оптимизации"

Выполнили студенты гр. 5030102/00201 Гвоздев С.Ю.,
(Алгоритм иммитации отжига)
Золин И.М.
(Формулировка задачи и её формализация, Муравьиный алгоритм)
Хламкин Е.М.
(Алгоритм)

Преподаватель: Родионова Е.А.

Санкт-Петербург

2022

Содержание

1 Введение

Задача коммивояжера (TSP -Travelling Salesman Problem) — одна из наиболее активно изучаемых задач вычислительной математики.

Эта задача состоит в том, чтобы найти кратчайший путь , по которому коммивояжер должен пройти через список городов и вернуться в исходный город. Приведен список городов и расстояние между каждой парой.

TSP полезен в различных приложениях в реальной жизни, таких как планирование или логистика. Например, менеджер концертного тура, который хочет запланировать серию выступлений для группы, должен определить кратчайший путь для тура, чтобы обеспечить сокращение транспортных расходов и не утомлять группу без необходимости.

Это NP-сложная задача. Проще говоря, это означает, что вы не можете гарантировать нахождение кратчайшего пути в разумные сроки. Однако это не уникально для TSP.

2 Формулировка задачи и её формализация

2.1 Формулировка задания

Рассматривается следующая формулировка задачи коммивояжера. Есть N городов, соединённых между собой односторонними дорогами. Коммивояжер выезжает из начального города, он должен посетить остальные N-1 городов и закончить свой путь в некотором заранее определенном городе(или вернуться обратно в стартовый город). Повторное посещение городов запрещено.

Дополнительно: каждый город характеризуется рейтингом, за заданное время нужно обойти города $\geq$ суммарного рейтинга.

2.2 Формализация задания

$n \geq 0$ - количество городов
$M_{n,n}$ - матрица времени, $m_{ij} \geq 0; i, j = \overline{1, n}$
$S \geq 0$ - ограничение по времени
C_n - массив рейтингов городов, $c_i \geq 0; i = \overline{1, n}$

$$\begin{cases} \sum_{i=1}^{k} c_i \to max; k = \overline{1, n} \\ \sum_{i=1}^{k} m_{ij} \leq S \end{cases} \tag{1}$$

(1) : Множество ограничений:

1. Функция цели (задача поиска максимума, суммарный рейтинг)
2. Ограничения (по времени)

3 Решение задачи

3.1 Алгоритм полного перебора

Метод полного перебора - самый наивный и времязатратный алгоритм поиска оптимального пути задачи коммивояжера с временными ограничениями, однако единственный, который всегда находит наиболее оптимальный путь.

Алгоритм метода прост: перебор осуществляется на заданных начальных данных (на- чальный город, ограничение по времени и псевдослучайная матрица времён) путём последо- вательной выборки наименьшего по индексу города из ещё непройденных. При включении

города в текущий путь к рейтингу пути добавляется значение рейтинга, соответствующее новому городу. Далее функция поиска подходящего города вызывается снова, уже считая

только что найденный город изначальным. Рекурсия происходит до тех пор, пока не кон- чится отведённое на переходы между городами время. Изначально максимальный рейтинг

пути задан как 0 (поэтому даже в худшем случае, когда ограничение времени меньше всех путей из города, рейтинг пути будет больше 0 и равен рейтингу 1-го города). Если за время,

меньшее ограничения, алгоритму удалось найти более благоприятный путь, переменная, от- вечающая за максимальный рейтинг пути, изменяет своё значение на свеженайденное, что

в конце концов и приводит к решению поставленной задачи.

3.2 Муравьиный алгоритм

3.2.1 Введение

Первоначально идею муравьиного алгоритма предложил Марко Дориго в 1992 году в его докторской диссертации, первый алгоритм был направлен на поиск оптимального пути в графе, основанном на поведении муравьев, ищущих путь между своей колонией и источником пищи для решения задач оптимизации.

В естественном мире муравьи бродят хаотично и, найдя пищу, возвращаются в свою колонию, прокладывая феромонные тропы. Другие же муравьи, находя тропы с феромонами, используют эту информацию, тем самым укрепляя феромонную тропу(положительная обратная связь).

(*):Чем короче путь до источника пищи, тем меньше времени понадобится на пермещение муравьям - следовательно, тем быстрее оставленные на нем следы будут заметны.

(**): В итоге, чем больше муравьев проходит по определенному пути, и чем этот путь короче, тем он становится более привлекательным для остальных муравьев.

3.2.2 Описание

Каждый муравей рассматривается как отдельный и независимый коммивояжер, решающий свою собственную проблему. В течение одной итерации муравей проходит весь маршрут коммивояжера.

"Случайность"реализует вероятностное правило, с пмощью которого обеспечивается положительная обратная связь: вероятность того, что ребро графа включено в маршрут муравья, пропорционально его значению феромона.

Чтобы имитировать поведение муравьев (**), объем виртуальных феромонов, размещенных на ребре графика, принимается обратно пропорциональным длине пути.

(*): Однако положительная обратная связь приводит к застою: в этом случае все муравьи выбирают один неоптимальный путь. Чтобы избежать этого, существует отрицательная обратная связь: через ферромон вводится испарение. Интенсивность испарения не должна быть слишком высокой; в противном случае область поиска сузится(ситуации, когда колония преждевременно “забывает” свой опыт, полученный в прошлом (потеря памяти)).

Для каждого муравья проход из города i в город j зависит от следующих трех компонентов: список запретов (tabu list) или память муравья, видимость и следы виртуальных феромонов.

Список запретов - это структура данных, которая сохраняет список уже посещенных городов, которые не следует посещать снова. Этот список увеличивается в размерах на каждом шаге и устанавливается равным нулю в начале каждой итерации алгоритма. Обозначение: $J_{i,k}$ - список городов, которые еще предстоит посетить k-ому муравью, расположенному в i-ом городе .

Видимость - является обратной величиной к расстоянию. $\eta_{i,j} = \frac{1}{D_{i,j}}$, $D_{i,j}$ - это расстояние между городами i и j. Видимость - это локальное статическое значение, отражающее эвристическое желание переехать из города i в город j: чем ближе город, тем сильнее желание его посетить.

Муравьиный алгоритм можно классифицировать как вероятностный, то есть он дает только приближенное решение, не гарантируя его оптимальности.

3.2.3 Формализация

Ниже приведён алгоритм:

1. Инициализация(создаем муравьёв и задаем начальные значения феромона)

2. Вероятность перемещения k-ого муравья на итерации t из города i в город j вычисляется по следующему вероятностно–пропорциональному правилу:

$$P_{i,j,k}(t) = \begin{cases} \frac{(\tau_{i,j}(t))^{\alpha}(\eta_{i,j})^{\beta}}{\sum_{l \in J_{i,k}}(\tau_{i,l}(t))^{\alpha}(\eta_{i,l})^{\beta}}, j \in J_{i,k} \\ 0, j \notin J_{i,k} \end{cases} \tag{2}$$

   где $\tau_{i,j}(t)$– уровень феромона, α, β– константные параметры(описывают вес феромонного следа и посещаемость при выборе маршрута).

   Когда $\alpha = 0$, выбирается ближайший город, что соответствует жадному алгоритму в классической теории оптимизации. Когда $\beta = 0$, учитывается только след феромона, что означает, что все муравьи выбирают один неоптимальный маршрут. Чтобы обеспечить хорошую динамику оптимизации, рекомендуется установить $\alpha \geq \beta$.

   Отметим, что (2) определяет вероятности выбора конкретного города. Сам выбор осуществляется по принципу "колеса рулетки": каждый город на нем имеет свой собственный сектор с площадью, соответствующей вероятности (2).

3. Феромон, откладываемый k-ым муравьём, использующим ребро (i,j)

$$\Delta\tau_{i,j,k}(t) = \begin{cases} \frac{Q}{L_k(t)}, (i,j) \in T_k(t) \\ 0, (i,j) \notin T_k(t) \end{cases} \tag{3}$$

   где $T_k(t)$ - это маршрут муравья k на итерации, $L_k(t)$ – длинна маршрута(цена текущего решения для k-ого муравья), а Q>0 – параметр, имеющий значение порядка цены оптимального решения.

4. Обновляем уровень феромона в соответствии с приведённой формулой

$$\tau_{i,j}(t+1) = (1-p)\tau_{i,j}(t) + \sum_{k=1}^{m} \Delta\tau_{i,j,k}(t) \tag{4}$$

где $p \in [0, 1]$ – коэффициент испарения, m - количество муравьев в колонии. На ранней стадии процесса оптимизации значение феромона на ребрах принимается равным небольшому положительному числу τ_0.

3.3 Алгоритм иммитации отжига

3.3.1 Описание

3.3.2 Формализация

3.4 Алгоритм поиска в глубину с возвратом

3.4.1 Описание

3.4.2 Формализация

4 Пример

5 Выводы

6 Библиографический список

1. Володина Е.В. Практическое применение алгоритма решения задачи коммивояжера/ Е.В. Володина, Е.А. Студентов
2. Громкович Ю. Алгоритмизация труднорешаемых задач. Часть I. Простые примеры и простые эвристики / Ю. Громкович, Б.Ф.Мельников
3. Громкович Ю. Алгоритмизация труднорешаемых задач. Часть II. Более сложные эвристики. / Ю. Громкович, Б.Ф.Мельников
4. Гудман С. Введение в разработку и анализ алгоритмов: учебное пособие / С. Гудман, С. Хидетниеми
5. Дулькейт В.И. Приближённое решение задачи коммивояжера методов рекурсивного построения вспомогательной кривой
6. Муравьиный алгоритм `https://habr.com/ru/post/105302/`
7. Муравьиный алгоритм(ч.2) `https://www.researchgate.net/publication/220203867_Ant_Algorithms_Theory_and_Applications`