Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра исследования операций

Шматенко Дарья Павловна

Задача коммивояжёра. Метод Литтла

4 курс, группа 411

Москва, 2020 г.

Постановка задачи

Задача коммивояжёра — одна из самых известных задач комбинаторной оптимизации, заключающаяся в поиске самого выгодного маршрута, проходящего через указанные города хотя бы по одному разу с последующим возвратом в исходный город. В условиях задачи указываются критерий выгодности маршрута (кратчайший, самый дешёвый, совокупный критерий и тому подобное) и соответствующие матрицы расстояний, стоимости и тому подобного. Как правило, указывается, что маршрут должен проходить через каждый город только один раз.

Проблему коммивояжёра можно представить в виде модели на графе, то есть, используя вершины и ребра между ними. Таким образом, вершины графа соответствуют городам, а рёбра (i, j) между вершинами i и j — пути сообщения между этими городами. Каждому ребру (i, j) можно сопоставить критерий выгодности маршрута $c_{ij} \geq 0$, который можно понимать как, например, расстояние между городами, время или стоимость поездки.

Гамильтоновым циклом (путём) называется маршрут, включающий ровно по одному разу каждую вершину графа. Контур —это конечный путь, у которого начальная вершина совпадает с конечной. Контур, проходящий через все вершины графа, называется гамильтоновым контуром.

В целях упрощения задачи и гарантии существования маршрута обычно считается, что модельный граф задачи является полностью связным, то есть, что между произвольной парой вершин существует ребро. В тех случаях, когда между отдельными городами не существует сообщения, этого можно достичь путём ввода рёбер с максимальной длиной. Из-за большой длины такое ребро никогда не попадет к оптимальному маршруту, если он существует.

Таким образом, решение задачи коммивояжёра — это нахождение гамильтонова цикла минимального веса в полном взвешенном графе.

Алгорит Литтла

Одним из известных ранних алгоритмов точного решения задачи коммивояжера для общего случая является алгоритм Литтла, основанный на методе ветвей и границ, который строит дерево решений для перебора вариантов маршрута (циклов обхода) с отсечением.

Вначале строится некоторая оценка снизу длины маршрута для множества всех гамильтоновых контуров. После этого множество всех гамильтоновых маршрутов разбивается на два подмножества. Первое подмножество состоит из гамильтоновых контуров, включающих некоторую дугу, а другое подмножество состоит из контуров, не включающих эту дугу. Для каждого из подмножеств по тому же правилу, что и для первоначального множества гамильтоновых маршрутов, определяется нижняя граница. Каждая новая нижняя граница оказывается не меньше нижней границы, определённой для всего множества. Сравнивая нижние границы, можно отделить подмножество гамильтоновых путей, внутри которого с большей вероятностью содержится оптимальный маршрут. Это подмножество по аналогичному правилу разбивается ещё на два, и вновь находятся изменённые нижние границы, и так поступают до тех пор, пока не останется единственный цикл. Процесс разбиения подмножеств сопровождается постороением дерева.

Получив некоторый гамильтонов цикл, просматривают оборванные ветви дерева и, если нижние границы подмножеств, соответствующие оборванным ветвям, окажутся меньше, чем длина найденного цикла, то эти ветви развивают по тому же правилу, пока не получат маршрута с меньшей длиной или не убедятся, что среди этих подмножеств не может содержаться подобного маршрута. Те же ветви, для которых нижняя оценка окажется больше длины полученного маршрута, исключают из рассмотрения.

Идея получения этой оценки связана с тем, что если решить задачу о коммивояжёре с некоторой матрицей расстоний, а затем из какой-нибудь строки или столбца этой матрицы вычесть произвольное положительное число, то решение задачи о коммивояжёре с этой изменённой матрицей расстояний совпадёт с прежним решением, а длина маршрута изменится на это же самое число. Если эту операцию проделать и для других строк и столбцов, то длина маршрута будет отличаться на сумму всех чисел, вычитаемых из строк и столбцов.

Опираясь на этот факт, осуществляют приведение матрицы: в каждой строке матрицы находят минимальный элемент и вычитают его из всех элементов этой строки. В результате по крайней мере один из элементов строки будет нуль. Так поступают со всеми строками. Полученная матрица называется приведённой по строкам.

После этого аналогичным образом осуществляется приведение полученной матрицы по столбцам.

Приведённая по строкам и столбцам матрица содержит по крайней мере один нуль в каждой строке и каждом столбце. Так как длина L_1 оптимального маршрута в задаче с приведённой матрицей отличается от длины L маршрута в задаче с неприведённой матрицей на сумму констант приведения h, то

$$L = L_1 + h.$$

В приведённой матрице все элементы неотрицательны, поэтому $L_1 \geq 0$, а сумма констант приведения h может служить нижней границей длины гамильтонова маршрута.

Априорное исключение какой-нибудь дуги (i, j) из маршрута выражается в замене соответствующего элемента $d_{i,j}$ в исходной (а также приведённой) матрице на ∞. В результате такой замены появляется возможность провести дополнительное приведение матрицы и улучшение оценки.

Априорное включение дуги (i, j) в маршрут автоматически ведёт к сокращению размеров матрицы (так как вычёркивается строка i и столбец j). Одновременно появляется возможность исключить одну из дуг.

Включение дуги (i, j) приводит к образованию связного пути из k в l. Допустимые маршруты должны проходить через все точки графа, поэтому запрещается включение в маршрут дуги (l, k), так как иначе образуется замкнутый контур, проходящий только через часть точек. В простейшем случае включение дуги (i, j) означает невключение дуги (j, i), т. е. $d_{i,j} = \infty$.

Сокращение размеров матрицы и исключение элемента (l, k) позволяет провести дополнительное приведение матрицы и тем самым улучшить нижнюю оценку.

Как выбрать дугу (i, j)? Наиболее вероятно, что в оптимальный маршрут входят дуги, для которых в приведённой матрице $d_{i,j} = 0$. Невключение в маршрут именно этих дуг резко увеличивает нижнюю оценку. Как уже отмечалось, исключая дугу (j, i), мы должны положить $d_{i,j} = \infty$ и, осуществляя приведение вновь полученной матрицы, улучшить оценку. Константами привидения являются минимальный элемент в i-й строке и такой же в i-м столбце. Поэтому наиболее резкое изменение оценки произойдёт в том случае, когда выбирается тот элемент матрицы расстояний, для которого:

1. $d_{i,j} = 0$;

2. после замены $d_{i,j} = 0$ на $d_{i,j} = \infty$ сумма минимальных элементов i-й строке и i-м столбце имеет наибольшее значение.

Детали реализации

На вход алгоритма подается квадратная матрица расстояний $C_{i,j}$ размером $n \times n$, все $c_{i,j} \geq 0$, причём диагональные элементы $c_{i,j} = \infty$. Строки и столбцы матрицы помечаются отрезками маршрута, включёнными в строящийся вариант маршрута. В самом начале все эти отрезки суть отдельные вершины. Отрезок маршрута, начинающийся из вершины k и заканчивающийся в вершине i, задаётся парой чисел (k, i). После того как выбрано ребро (i, j), находящееся на пересечении строки, помеченной отрезком (k, i), и столбца, помеченного отрезком (j, l), для вариантов маршрута 1-й группы эти два отрезка объединяются. Для этого из матрицы расстояний удаляется строка (k, i) и столбец (j, l), в результате чего число строк и столбцов матрицы уменьшается на 1. Строка, помеченная отрезком, заканчивающимся в вершине l, и столбец, помеченный отрезком, начинающимся с вершины k, помечаются отрезком (k, l). Кроме того, элемент матрицы на их пересечении заменяется величиной ∞. Это делается для того, чтобы предотвратить в дальнейшем выбор ребра графа (l, k), замыкающего локальный цикл, не включающий все вершины.

Для вариантов маршрута 2-й группы выбранное ребро (i, j) в матрице заменяется величиной ∞, что предотвращает в последующем выбор этого ребра для маршрута. При этом размер матрицы не изменяется.

Когда размер матрицы уменьшится до 2×2, если при этом оценка снизу окажется меньше стоимости ранее найденного наилучшего маршрута, то строится окончательный замкнутый маршрут путем включения в него как ребра (i, j), так и ребра (l, k). Стоимость нового маршрута совпадает с его оценкой снизу, и он запоминается как наилучший маршрут среди всех ранее просмотренных.

Алгоритм Литтла просматривает строящееся таким образом дерево решений вглубь, в котором сначала делается переход к первой группе вариантов маршрута, а после их просмотра — ко второй группе.