

Пример. Дан взвешенный оргграф (рис. 4.52). Найти кратчайший гамильтонов цикл.

Решение. Способ 1. Рассмотрим алгоритм ближайшего соседа (Nva — Nearest vertex add)¹. Начнем движение по графу из произвольной вершины, например из вершины a . Выбирая из двух возможных

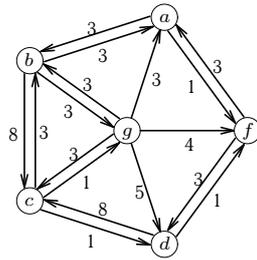


Рис. 4.52

дуг, $a \rightarrow f$ и $a \rightarrow b$, короткую дугу, $a \rightarrow b$, длиной 1, далее до вершины c двигаемся без разветвлений: $a \rightarrow f \rightarrow d \rightarrow c$. Из c направляемся по более короткой дороге в g , затем в b и возвращаемся в a . Суммарная длина цикла равна $1 + 3 + 8 + 1 + 3 + 3 = 19$. Алгоритм ближайшего соседа не гарантирует оптимального решения. Решение зависит от выбора начальной вершины и от выбора направления движения при наличии

двух и более направлений одинакового веса. Проверим другую вершину в качестве начальной.

Если начинать движение из b в сторону a по дуге весом 3, то цикл $b \rightarrow a \rightarrow f \rightarrow d \rightarrow c \rightarrow g \rightarrow b$ будет совпадать с найденным, отличаясь только началом отсчета. При выборе направления $b \rightarrow g$, также весом 3, получим цикл $b \rightarrow g \rightarrow c \rightarrow d \rightarrow f \rightarrow a \rightarrow b$, имеющий вес $3 + 3 + 1 + 1 + 3 + 3 = 14$. Этот цикл, наименьший по весу, можно принять за окончательный ответ. В действительности, суммарный вес 14 является наименьшим для этой задачи.

Известна оценка алгоритма ближайшего соседа [2] для графа, веса которого удовлетворяют неравенству треугольника (см. с. 81):

$$c_{Nva} \leq 0,5(\lceil \ln n \rceil + 1)c_{opt},$$

где c_{opt} — вес оптимального маршрута. В нашем случае коэффициент в этой оценке $0,5(\lceil \ln n \rceil + 1) = 1$.

Соответствующая программа для **Maple** дана на с. 145.

Способ 2. Рассмотрим алгоритм ближайшей вставки (Nvi — Nearest insert) [2]. Построение цикла начнем с любой вершины, например c . Присоединим к c вершину, образующую вместе с ней цикл. Из трех возможных вариантов, $c \rightarrow d \rightarrow c$, $c \rightarrow g \rightarrow c$ и $c \rightarrow b \rightarrow c$, возьмем самый короткий, $c \rightarrow g \rightarrow c$, длиной 9 (рис. 4.53). Конечно, можно выбрать и вариант $c \rightarrow b \rightarrow c$ такой же длины. На следующем шаге можно добавить либо вершину d , вставляя дуги $g \rightarrow d$ и $d \rightarrow c$ вместо

¹Аналогичный по стратегии алгоритм использовался для определения остова наименьшего веса (см. с. 79).

дуги $g \rightarrow c$, либо b уже с двумя вариантами циклов: $g \rightarrow c \rightarrow b \rightarrow g$ и $g \rightarrow b \rightarrow c \rightarrow g$. Выбираем цикл наименьшей длины: $g \rightarrow b \rightarrow c \rightarrow g$ (рис. 4.54). Аналогично последовательно вставляем в цикл вершины d , f , a (рисунки 4.55–4.57). В результате получаем цикл длиной $3 + 3 + 1 + 3 + 8 + 3 = 21$.

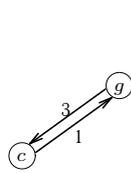


Рис. 4.53

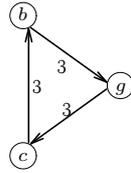


Рис. 4.54

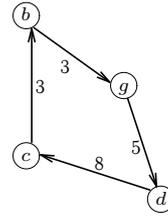


Рис. 4.55

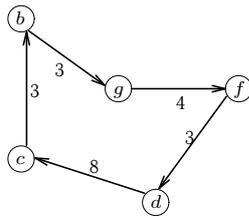


Рис. 4.56

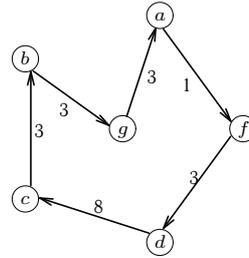


Рис. 4.57

Результат далек от оптимальной длины 14. В рамках этого алгоритма можно добиться меньшей длины, меняя начальную вершину и направления в точках ветвления решения.

Известна оценка алгоритма ближайшей вставки [2] для графа, веса которого удовлетворяют неравенству треугольника (см. с. 81):

$$c_{Nvi} \leq 2c_{opt},$$

где c_{opt} — вес оптимального маршрута.

Способ 3. Одним из методов искусственного интеллекта является муравьиный алгоритм Марко Дориго¹. Основная идея алгоритма подсмотрена в природе и имитирует движение колонии муравьев. По форме этот алгоритм похож на жадный Nva (способ 1) и в некоторой степени является его обобщением. Если в алгоритме ближайшего соседа выбор дальнейшего пути производится, исходя из минимального расстояния до очередной вершины, то здесь выбором управляет случайная функция, направляющая движение от текущего положения с

¹Marco Dorigo [33].

большой вероятностью в вершину j с наибольшим значением некоторой функции $P_{ij,k}$ (где i — номер вершины, в которой производится выбор, k — номер муравья, движущегося по дугам графа). Как и в Nva, во время движения создается список пройденных вершин, что позволяет избежать преждевременного заикливания. Приведем вид функции, управляющей переходом из данной вершины i в вершину j :

$$P_{ij,k} = \frac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_m \tau_{im}^\alpha \eta_{im}^\beta}, \quad (4.4)$$

где τ_{ij} — количество феромона (pheromon), оставленного муравьями на дуге $[i, j]$; η_{ij} — величина, обратная весу (длине) дуги $[i, j]$; α, β — эмпирические коэффициенты. Функция $P_{ij,k}$ подсказывает муравью номер вершины j , в которую он должен направиться. В знаменателе (4.4) стоит нормирующий коэффициент, такой, что $0 \leq P_{ij,k} \leq 1$. Индекс m в сумме пробегает по всем непройденным вершинам, смежным с i . В реальности муравей оставляет след (феромон) во время прохождения пути, и чем чаще он возвращается в исходную точку (а это возможно, если он выбирает оптимальные пути), тем четче след. В математической же модели функция τ_{ij} увеличивается только по завершении маршрута на величину, обратно пропорциональную длине маршрута. При $\alpha = 0$ алгоритм совпадает с Nva — муравей руководствуется только длиной пути. При $\beta = 0$ основой для выбора пути является только опыт (количество феромона, или «глубина следа») предыдущих муравьев-исследователей. Важно отметить еще одно отличие от алгоритма Nva. Выбор пути производится не по максимуму функции $P_{ij,k}$, а случайным образом, но на случай, конечно, влияет значение $P_{ij,k}$. Поясним это на примере. Пусть муравей k подошел к некоторой вершине 8 и обнаружил, что перед ним 7 возможных путей к семи вершинам (на уже пройденные он внимания не обращает). Куда идти? Муравей доверяется случаю. Он «пускает рулетку» (рис. 4.58). В какой

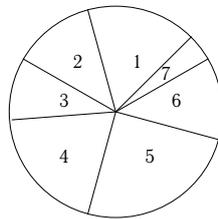


Рис. 4.58

сектор «шарик» закатится, туда и идти. Однако рулетка, размеченная функцией $P_{8j,k}$, $j = 1, \dots, 7$, имеет неравные сектора. Чем ближе

вершина и чем глубже туда след, тем больше сектор. Таким образом, муравей использует и опыт предшественников (τ_{ij}), и здравый смысл (η_{ij}), и случайный фактор, т.е. все как в жизни.

Для того чтобы не пропустить оптимальное решение, в муравьином алгоритме предусмотрено «испарение» следа. Это достигается введением коэффициента p в итеративной формуле $\tau_{ij} = (1 - p)\tau_{ij}$, применяющейся после каждого цикла обхода графа.

В алгоритме действует целая колония муравьев. Математически это означает, что в каждом цикле обхода движение производится из разных вершин независимым образом.

Здесь приведен самый простой вариант алгоритма. Алгоритм может быть улучшен. Для ускорения сходимости иногда вводят так называемых *элитных* муравьев [33]. В [7] приведены наилучшие комбинации параметров α и β .

Соответствующая программа для **Maple** дана на с. 148.

Способ 4. Алгоритм отжига. Этот алгоритм, как и муравьиный алгоритм, относится к вероятностным методам решения. Ключевым моментом в таких подходах является случайный выбор одного из нескольких возможных решений вместо анализа каждого. Это позволяет сократить время счета, а время счета в некоторых задачах на графах имеет принципиальное значение. Простой перебор для задач, сложность которых (время счета) растет по показательному или факториальному закону в зависимости от порядка графа, может даже для небольших графов оказаться недопустимо длительным. Оценка этой характеристики в каждом приложении своя. Например, для мобильного робота¹, выполняющего маневр в поиске оптимального решения, бортовой компьютер должен решать задачу коммивояжера за доли секунды. Прямой же перебор вариантов при пяти-десяти пунктах назначения возможен только за несколько минут, что в данном случае никак неприемлемо. При оптимизации проектируемых в лабораторных условиях тепловых, газовых, электрических или транспортных сетей, как правило, с сотней или более вершин (узлов), проект может затянуться даже на многие годы. Именно поэтому актуальны вероятностные подходы, дающие, может быть, не самые точные, но вполне допустимые решения. Рассмотрим метод отжига, обычно изучаемый в курсах теории искусственного интеллекта.

Образное название лишь отчасти передает содержание и связывается с процессом образования структуры металла при нагревании и дальнейшем охлаждении. Известно, что только контролируемое охлаждение приводит к желаемой структуре металла. При большей температуре степень свободы частиц (в нашем случае — количество вариантов

¹Мобильные роботы и мехатронные системы: Материалы научной школы – конференции. — М.: Изд-во Моск. ун-та, 2000.

решения) больше, при меньшей — меньше. Если дать возможность алгоритму случайно выбирать решение, оптимальное на каждом шаге (жадный алгоритм), то можно пропустить ход, не лучший локально, но дающий в результате более оптимальное решение.

В методе отжига очередной порядок следования по маршруту между городами выбирается случайно, небольшим изменением предыдущего решения — перестановка двух случайно выбранных городов в маршруте следования. Если полученный маршрут лучше всех существовавших ранее, то этот маршрут берется за очередной ¹. Если маршрут хуже, то самый простой вариант — это не брать его (зачем ухудшать решение?). Однако так решение может закатиться в один из локальных минимумов, которыми изобилуют подобные задачи (рис. 4.59). Именно

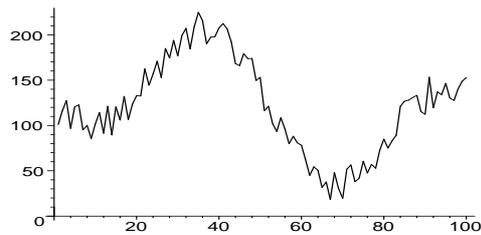


Рис. 4.59

поэтому плохому решению надо дать шанс. Шанс этот (или, правильнее, вероятность) рассчитывается по формуле

$$P = \exp(-\Delta L/T),$$

где ΔL — положительная разность между качеством тестируемого и ранее полученного оптимального решений, T — некоторый постоянно уменьшающийся параметр (условно — температура). В случае задачи коммивояжера качество оценивается длиной маршрута. Очевидно, что чем меньше текущее решение портит минимум и чем больше температура, тем больше вероятность принятия «пробного» выхода из локального минимума.

Снижение температуры обычно производится по формуле $T_{k+1} = \alpha T_k$, где $0 < \alpha < 1$.

Соответствующая программа для **Maple** дана на с. 152.

¹Иногда полученный вариант сравнивается с предыдущим.