Восстановление кратчайшего пути

Для того, чтобы иметь возможность восстановить кратчайший путь между любыми двумя вершинами необходимо выполнить предварительную работу:

1. До выполнения алгоритма Флойда-Уоршалла проинициализировать матрицу p, в которой в $p[u, v]$ будет храниться предпоследняя вершина на кратчайшем пути из u в v. До основной итераций необходимо проинициализировать $p[i, j] = i$ для каждого ребра $(i, j) \in E$ и $p[i][i] = i$ для каждой вершины $i \in V$. В иных случая $p[i, j]$ равно пустому значению (null);

2. если в основном цикле оказалось, что $d_{ij}^{k-1} > d_{ik}^{k-1} + d_{kj}^{k-1}$, то значит с добавлением в рассмотрение промежуточной вершины k обнаружился более коротки путь и поэтому необходимо обновить значение $p[i][j]$ до $p[k][j]$.

После заполнения матрицы p, восстановление кратчайшего пути из u в v выполняется следующим образом:

1. если $p[u][v]$ пустое, то значит нет пути из u в v;

2. если не пустое, то в обратном порядке проходим по всему пути от v до u: сначала добавляем v в текущий ответ r, далее, находясь в текущей вершине $j \neq u$, добавляем в r вершину $p[u][j]$ и переходим к вершине $p[u][j]$;

3. заканчиваем в момент, когда $j = u$. Возвращаем r в обратном порядке.

9.3.7 Задача о бродячем торговце

Задача о бродячем торговце или **задача о коммивояжере** (англ. travelling salesman problem, TSP) звучит следующим образом: в стране есть N городов, любые два соединены дорогой. Поездка по каждой дороге стоит определенную сумму. Необходимо объехать все N городов ровно по одному разу и потратить минимальное количество средств. С точки зрения теории графов задача о коммивояжере состоит в поиске во взвешенном полном неорграфе гамильтонова цикла наименьшего веса.

Замечание 79. Эта задача является NP-полной и существование ее быстрого решения маловероятно (см. Кормен et al. [2010]).

Оптимальная структуры оптимального решения

Оптимальное решение может быть рассмотрено как путь, начинающийся в произвольной вершине v_0 (должны обойти все вершины, поэтому не важно какую считать первой) и в ней же заканчивающийся.

Обозначим за $D(S, v)$ решение подзадачи поиска кратчайшего пути, начинающегося в v_0, проходящего через все вершины из множества S и заканчивающийся в v. Тогда кратчайший путь будет состоять из двух частей:

1. последнее ребро из некоторой вершины $w \in S$ в конечную вершину v;

2. кратчайший (если он не кратчайший, то легко дойти до противоречия с определением $D(S, v)$) путь из v_0 в w, проходящего через все вершины в $S \setminus \{v\}$.

Наперед знать какая вершина будет предпоследней в оптимальном пути из v_0 в v не получится, поэтому надо будет посмотреть все ребра w, v при $w \in S \setminus \{v\}$ и для каждой из них решить соответствующую подзадачу поиска кратчайшего пути. Таким образом, мы показали наличие оптимальной подструктуры.

Замечание 80. Заметим, что количество подзадач равно количеству возможных подмножеств множества вершин, умноженному на количество вершин n, то есть $2^n n$.

Рекурсивное определение оптимального решения

Пусть стоимости проезда по дорогам (веса в графе) заданы весовой функцией $c : E \to R$. На основе предыдущего анализа рекурсивное соотношение будет выглядеть следующим образом:

$$D(S, v) = \begin{cases} c(v_0, v), & S = \{v\} \\ +\infty, & v \notin S \\ \min\limits_{w \in S \setminus \{v\}} (D(S \setminus \{v\}, w) + c(w, v)), & \text{иначе} \end{cases} \tag{9.8}$$

Вычисление решения методом восходящего анализа

На основе рекурсивного соотношения 9.8 и входных данных (вершины V и весовая функция c) составим алгоритм поиска оптимального обхода:

1. Задаем произвольную стартовую вершину v_0; инициализируем матрицу D размера $2^n \times n$ значениями $+\infty$; для восстановления оптимального обхода введем матрицу P размера $2^n \times n$, где в $P[S][v]$ хранить последние ребра в оптимальных путях до v с обходом всем вершин в S;

2. Для всех вершин $v \in V$ записываем в $D[\{v\}][v]$ значение $c(v_0, v)$;

3. Далее итерируемся по всем возможным (от 2 до n) размерам i подмножеств вершин S, смотрим на каждое подмножество S: $|S| = i$ и на каждую возможно вершину $v \in S$. Для это вершины v находим такую $w \in S \setminus \{v\}$, что достигается минимум из соотношения 9.8. В $P[S][v]$ записывается ребро (u, v).

4. После всех вычислений (заполнены матрицы D и P) составим оптимальный обход: необходимо начать с ребра (u, v), находящегося в $P[V][v_0]$ и далее в цикле обнаруживаем предшествующую вершину j в цикле и находим ребро из оптимального обхода, ведущее в нее, и переходим к рассмотрению вершины на другом конце (на забывая выкидывать уже просмотренные вершины). Для получения стоимость поездки необходимо просто складывать веса всех ребер при обходе.

Замечание 81. Сложность этого алгоритма $O(2^n n^2)$, т.к. для каждой из подзадач мы вызываем еще дополниться не более, чем n мелких подзадач.

Пример 55. Применим описанный алгоритм решения задачи о коммивояжере к задаче, заданной графом с Рис. 9.9. В качестве стартовой вершины выберем $v_0 = 0$. Заполненные матрицы D и P представлены на Рис. 9.10. Для составления

оптимального маршрута мы пройдем последовательность шагов $P[0123][0] = (3, 0)$; $P[123][3] = (1, 3)$; $P[12][1] = (2, 1)$; $P[2][2] = (0, 2)$.

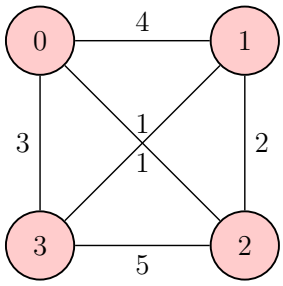

Рис. 9.9: Пример задачи о коммивояжере. Стартовая вершина 0.

	0	1	2	3
()	$+\infty$	$+\infty$	$+\infty$	$+\infty$
0	$+\infty$	$+\infty$	$+\infty$	$+\infty$
1	$+\infty$	4	$+\infty$	$+\infty$
2	$+\infty$	$+\infty$	1	$+\infty$
3	$+\infty$	$+\infty$	$+\infty$	3
01	8	$+\infty$	$+\infty$	$+\infty$
02	2	$+\infty$	$+\infty$	$+\infty$
03	6	$+\infty$	$+\infty$	$+\infty$
12	$+\infty$	3	6	$+\infty$
13	$+\infty$	4	$+\infty$	5
23	$+\infty$	$+\infty$	8	6
012	7	6	9	$+\infty$
013	8	10	$+\infty$	11
023	9	$+\infty$	7	5
123	$+\infty$	7	6	4
0123	7	6	9	7

(a) Матрица D

	0	1	2	3
()	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
0	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
1	$\varnothing$	$(0, 1)$	$\varnothing$	$\varnothing$
2	$\varnothing$	$\varnothing$	$(0, 2)$	$\varnothing$
3	$\varnothing$	$\varnothing$	$\varnothing$	$(0, 3)$
01	$(1, 0)$	$\varnothing$	$\varnothing$	$\varnothing$
02	$(2, 0)$	$\varnothing$	$\varnothing$	$\varnothing$
03	$(3, 0)$	$\varnothing$	$\varnothing$	$\varnothing$
12	$\varnothing$	$(2, 1)$	$(1, 2)$	$\varnothing$
13	$\varnothing$	$(3, 1)$	$\varnothing$	$(1, 3)$
23	$\varnothing$	$\varnothing$	$(3, 2)$	$(2, 3)$
012	$(1, 0)$	$(0, 1)$	$(0, 2)$	$\varnothing$
013	$(3, 0)$	$(0, 1)$	$\varnothing$	$(0, 3)$
023	$(3, 0)$	$\varnothing$	$(0, 2)$	$(0, 3)$
123	$\varnothing$	$(3, 1)$	$(1, 2)$	$(1, 3)$
0123	$(3, 0)$	$(3, 1)$	$(0, 2)$	$(1, 3)$

(b) Матрица P

Рис. 9.10: Пример работы алгоритма динамического программирование решения задачи о коммивояжере. Оптимальный путь будет $(0, 2, 1, 3, 0)$.

9.4 Упражнения

Задача 88. В задаче о минимальном размене $N = 24$ и номиналы $\{w_i\} = (1, 3, 6, 7, 11)$. Найдите мощность минимального размена и входящие в него номиналы.

Задача 89. В задаче о минимальном размене $N = 24$ и номиналы $\{w_i\} = (2, 5, 8, 10, 13)$. Найдите мощность минимального размена и входящие в него номиналы.

Задача 90. Пусть дан рюкзак емкости $W = 20$ и предметы с парами (вес, ценность) из таблицы: