УДК 519.2

АЛГОРИТМ ДЕЙКСТРЫ

Микушина К.С., студентка гр. ПСб-251, I курс
Научный руководитель: Липина Г.А., старший преподаватель
Кузбасский государственный технический
университет имени Т.Ф. Горбачева, г. Кемерово

Каждый из нас хоть раз пользовался онлайн картой на своем смартфоне, чтобы проложить маршрут до места назначения, но мало кто знает, что чтобы предложить вам оптимальный путь, в приложении используется алгоритм, придуманный более полувека назад. Алгоритм Дейкстры – алгоритм на графах, разработанный голландским учёным Эдсгером Дейкстрой в 1959 году, который находит кратчайшие пути от одной из вершин графа до всех остальных. Несмотря на то, что алгоритм был разработан ещё во время становления вычислительной техники, он не потерял актуальность до сих пор и продолжает использоваться во многих сферах. Например, таких как: логистика, робототехника, интернет, навигация и т.д.

Основная идея алгоритма заключается в том, что алгоритм последовательно находит вершины, для которых кратчайшее расстояние от начальной точки окончательно определено. Из необработанных вершин выбирается та, чья текущая оценка расстояния минимальна. Эта оценка закрепляется за вершиной. После этого выполняется релаксация: проверяется, позволяет ли путь через добавленную вершину сократить уже вычисленные расстояния до ее соседей.

Чтобы разобраться как работает данный алгоритм для начала введем формальные понятия дискретной математики.

Граф $G = (V, E)$

где V – множество вершин, E – множество ребер.

Ребра графа бывают направленными и ненаправленными. Первые формируют ориентированный граф, вторые – неориентированный. При поиске пути чаще работают с неориентированными графами т.е. дорога соединяет два перекрестка, и двигаться можно в обе стороны. Хотя важно заметить, что алгоритм Дейкстры справляется с обоими типами одинаково эффективно.

Для моделирования же реальных расстояний, временных затрат или стоимости используют взвешенный граф. Каждому его ребру приписывается числовое значение, называемое весом ребра. Формально это задается функцией веса: $w: E \rightarrow R$. Для ребра $(u, v) \in E$, его вес обозначается как $w(u, v)$. В неори-

ентированном графе вес симметричен: w(u, v) = w(v, u) т.е. не зависит от направления движения.

Путь в графе от вершины A до вершины B – это конечная последовательность вершин P.

$P = (v_0, v_1, \ldots, v_k)$ при этом:

во-первых, v_0 = A (первая вершина и начало пути) и v_k = B (последняя вершина и конец пути),

во-вторых, для каждого i = 1, …, k пара (v_{i-1}, v_i) является ребром графа, т.е.

$(v_{i-1}, v_i) \in E$

Вес пути (P) определяется как сумма весов всех ребер:

$$w(P) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

где k обозначается как длина пути (количество ребер), а (v_{i-1}, v_i) – как вес ребра между соседними вершинами.

Кратчайшим путем (P*), называется путь, имеющий минимальный вес среди остальных путей и формально обозначается следующим образом:

$$w(P^*) = \min_{P: s \to t} w(P)$$

Важно заметить, что может существовать несколько кратчайших путей, имеющих одинаково минимальный вес.

Теперь, вооружившись всеми необходимыми определениями, рассмотрим работу алгоритма на примере.

Рассмотрим взвешенный граф G с вершинами V = {A, B, C, D, E} и весовой функцией w: $E \to R^+$. Известно, что вес ребер AB, AC, BC, BD, CD, CE, DE равен 4, 2, 1, 5, 8, 10, 2 соответственно. Необходимо найти кратчайший путь из вершины A в вершину E.

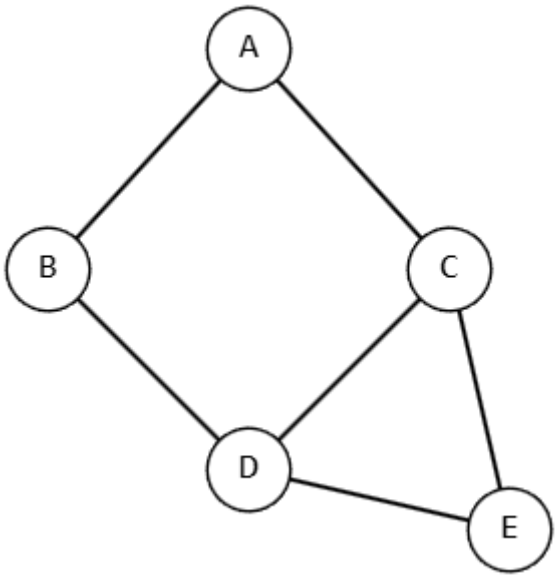

Рисунок 1 – Граф.

S – вершины, для которых кратчайшее расстояние уже найдено окончательно.

T – ещё не обработанные вершин.

Шаг 1.

В множестве T выбираем вершину с наименьшей меткой. Это вершина A (d(A) = 0), переносим её в S: S = {A}, T = {B, C, D, E}.

Выполняем релаксацию для всех рёбер, идентичных A:

Для ребра (A, B): d(A) + w(A, B) = 0+4 = 4. Так как 4 < d(B) = +∞, следовательно теперь d(B) = 4.

Для Ребра (A, C): 0+2 = 2 < +∞, d(C) = 2.

Текущее состояние после шага: d(A) = 0, d(B) = 4, d(C) = 2, d(D) = +∞, d(E) = +∞.

Шаг 2.

В T = {B, C, D, E} наименьшая метка у вершины C (d(C) = 2).

Переносим C в S: S = {A, C}, T = {B, D, E}.

Релаксация по рёбрам из C (вершины A, B, D, E):

Для ребра (C, B): d(C) + w(C, B) = 2+1 = 3. Так как 3 < d(B) = 4, d(B) = 3.

Для ребра (C, D): 2+8=10<+∞, d(D) = 10.

Для ребра (C, E): 2+10=12<+∞, d(E) = 12.

Состояние после шага: d(A) = 0, d(B) = 3, d(C) = 2, d(D) = 10, d(E) = 12.

Шаг 3.

В T = {B, D, E} минимальная метка у B(d(B)=3). Переносим B в S: S = {A, C, B}, T = {D, E}.

Релаксация по рёбрам из B (кроме уже обработанных A и C):

Для Ребра (B, D): d(B) + w(B, D) = 3+5 = 8. Так как 8 < d(D) = 10, d(D) = 8.

Состояние после шага: d(A) = 0, d(B) = 3, d(C) = 2, d(D) = 8, d(E) = 12.

Шаг 4.

В T = {D, E} наименьшая метка у D (d(D) = 8).

Переносим D в S: S = {A, C, B, D}, T = {E}.

Релаксация по рёбрам из D (кроме B и C):

Для ребра (D, E): d(D) + w(D, E) = 8+2 = 10. Так как 10 < d(E) = 12, d(E) = 10.

Состояние после шага: d(A) = 0, d(B) = 3, d(C) = 2, d(D) = 8, d(E) = 10.

Шаг 5.

В множестве T осталась только вершина E с меткой d(E) = 10. Это искомая целевая вершина. Ее метка минимальна среди всех вершин T, поэтому E переходит в множество S. Множество S теперь содержит все вершины графа. Алгоритм завершен. Таким образом, был найден кратчайший путь из вершины A в вершину E: A→C→B→D→E.

Список литературы:

1. Кормен, Т. Алгоритмы: построение и анализ/ Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. – 3-е изд. – Москва: Вильямс, 2013. – 1328 с.
2. Локшин, А. А. Элементарное введение в теорию графов: учебное пособие/ А. А. Локшин. – 2-е изд., испр. и доп. – Москва: МАКС Пресс, 2024. – 146 с.