**Федеральное государственное образовательное учреждение**

**Высшего профессионального образования**

**Национальный исследовательский технологический университет «МИСиС»**

**Институт информационных технологий и компьютерных наук (ИТКН)**

**Кафедра автоматизированных систем управления (АСУ)**

**Курсовая работа**

**по курсу «Алгоритмы дискретной математики»**

**Тема:**

**«Алгоритм Форда-Беллмана»**

­­­

**Выполнил: студент 2 курса,**

**гр. БИВТ-21-9 Мангушев И. Н.**

**Руководитель: Валова А. А.**

**Москва 2023**

Оглавление

[**Введение 3**](#_Toc136903047)

[**Постановка задачи 4**](#_Toc136903048)

[**Теоретическая часть 5**](#_Toc136903049)

[Описание 5](#_Toc136903050)

[Основная идея 5](#_Toc136903051)

[Асимптотическая оценка 6](#_Toc136903052)

[Сравнение алгоритма с другими 6](#_Toc136903053)

[Примеры работы 7](#_Toc136903054)

[**Заключение 8**](#_Toc136903055)

[**Список литературы 8**](#_Toc136903056)

[**Программные приложения 8**](#_Toc136903057)

Введение

Алгоритм Форда-Беллмана является одним из классических алгоритмов нахождения кратчайших путей в графе. Он был разработан в 1956 году американскими математиками Лестером Фордом и Ричардом Беллманом. Алгоритм Форда-Беллмана широко применяется в различных областях, таких как транспортное планирование, логистика, маршрутизация сетей и другие, где требуется оптимальное нахождение пути с минимальной стоимостью.

В данной курсовой работе будет рассмотрено применение алгоритма Форда-Беллмана в контексте логистической отрасли. Логистические компании активно используют этот алгоритм для решения задач оптимизации маршрутов доставки грузов, планирования ресурсов и управления логистической сетью. Рассмотрение применения алгоритма Форда-Беллмана в логистике позволит оценить его эффективность, преимущества и ограничения в контексте реальных бизнес-задач.

Целью данной курсовой работы является исследование и анализ алгоритма Форда-Беллмана в контексте логистической отрасли с целью определения его эффективности и применимости для решения конкретных задач логистики. Результаты работы позволят логистическим компаниям принять информированное решение о выборе оптимального алгоритма для своих потребностей в области оптимизации маршрутов и планирования ресурсов.

Основная задача, решаемая алгоритмом Форда-Беллмана, состоит в нахождении кратчайшего пути от одной вершины графа до всех остальных вершин. В отличие от других алгоритмов, таких как алгоритм Дейкстры или алгоритм А\* (A-star), алгоритм Форда-Беллмана способен работать с графами, содержащими ребра с отрицательными весами.

Алгоритм Форда-Беллмана основан на принципе динамического программирования. Он строит кратчайшие пути поэтапно, начиная с самых коротких путей и постепенно увеличивая их длину. Алгоритм повторяет этот процесс до тех пор, пока все возможные пути не будут исследованы и длины путей не стабилизируются.

В работе будет представлен обзор алгоритма Форда-Беллмана, его основные шаги и принципы работы. Будут рассмотрены различные варианты применения алгоритма в логистической отрасли, а также представлены практические примеры и исследования, подтверждающие его эффективность.

Постановка задачи

Формальная постановка задачи:

Нам задается ориентированный граф G(V, E), где V представляет собой множество вершин графа, а E - множество ребер, и каждое ребро (u, v) имеет свой вес w(u, v), который может быть положительным, отрицательным или нулевым. Одна из вершин графа, называемая источником или начальной вершиной s. После определения стартовой вершины алгоритм инициализирует расстояние от источника s до v как бесконечность, кроме расстояния до источника s, которое устанавливается как 0. Далее происходит перебор по каждому ребру (u, v) в графе и обновляется расстояние до вершины v, если расстояние от источника s до вершины u плюс вес ребра (u, v) меньше текущего расстояния до вершины v. Повторение шагов происходит |V|-1 раз. Одновременно с прохождением графа проверяется наличие отрицательных циклов. Если в результате еще одной итерации обновления расстояний происходят изменения, то граф содержит отрицательный цикл. В этом случае нельзя определить кратчайшие пути, так как они не существуют. Полученные расстояния от источника s до всех остальных вершин графа представляют собой длины кратчайших путей от источника до каждой вершины. Также можно сохранить информацию о предшествующих вершинах на кратчайших путях для восстановления самих путей.

Неформальная постановка задачи:

Задается набор мест на карте, между которыми необходимо найти кратчайшие пути. Для каждой пары мест определены время пути между ними. Нам требуется найти наименьшее время перемещения между всеми парами мест. Нам требуется инициализировать начальное место, от которого будет производиться поиск кратчайших путей. Произвести проход по всем возможным путям, перебирая все комбинации мест, и вычислять время каждого пути. Обновить наименьшее время для каждой пары мест при обнаружении пути с меньшем временем. Повторять шаги, пока не будет пройдено достаточное количество итераций или не будет достигнута сходимость, то есть не будет изменений во времени. Полученные наименьшие стоимости или расстояния представляют собой оптимальные значения для перемещения между всеми парами мест.

Теоретическая часть

Описание

Алгоритм носит имя двух американских учёных: Ричарда Беллмана (Richard Bellman) и Лестера Форда (Lester Ford). Форд фактически изобрёл этот алгоритм в 1956 г. при изучении другой математической задачи, подзадача которой свелась к поиску кратчайшего пути в графе, и Форд дал набросок решающего эту задачу алгоритма. Беллман в 1958 г. опубликовал статью, посвящённую конкретно задаче нахождения кратчайшего пути, и в этой статье он чётко сформулировал алгоритм в том виде, в котором он известен нам сейчас.

**Алгоритм Форда-Беллмана** позволяет найти кратчайшие пути из одной вершины графа до всех остальных, даже для графов, в которых веса ребер могут быть отрицательными. Тем не менее, в графе не должно быть циклов отрицательного веса, достижимых из начальной вершины, иначе вопрос о кратчайших путях является бессмысленным. При этом алгоритм Форда-Беллмана позволяет определить наличие циклов отрицательного веса, достижимых из начальной вершины.

Алгоритм использует динамическое программирование. Введем функцию динамического программирования:

F[k][i] — длина кратчайшего пути из начальной вершины до вершины i, содержащего не более k ребер.

Начальные значения зададим для случая k=0. В этом случае F[0] [start] = 0, а для всех остальных вершин i F[0][i] = INF.

Далее будем вычислять значения функции F увеличивая число ребер в пути k, то есть вычислим кратчайшие пути, содержащие не более 1 ребра, кратчайшие пути, содержащие не более 2 ребер и т. д. Если в графе нет циклов отрицательного веса, то кратчайший путь между любыми двумя вершинами содержит не более ребра (- число вершин в графе), поэтому нужно вычислить значения F[n-1] [i], которые и будут длинами кратчайших путей от вершины start до вершины i). Целевая функция:

Основная идея

Основная идея алгоритма Форда-Беллмана будет заключаться в следующих шагах:

1. Устанавливаются начальные значения расстояний от начальной вершины до всех остальных вершин графа. Расстояние до начальной вершины устанавливается равным 0, а расстояние до всех остальных вершин - бесконечности.
2. Проход по всем ребрам графа и обновление расстояний до вершин, если найден путь с меньшей стоимостью.
3. Пункт 2 повторяется для всех вершин графа |V|-1 раз, где |V| - количество вершин графа. Это позволяет учитывать пути с более чем одним ребром и обновлять оценки кратчайших путей.
4. Если после |V|-1 итераций происходит изменение расстояний, то это указывает на наличие отрицательного цикла в графе. В этом случае кратчайшие пути не могут быть определены, так как цикл может быть проходом с отрицательной суммой весов, и алгоритм продолжит обновлять расстояния бесконечно.

Реализация кода в данной работе:

// Функция для поиска кратчайших путей от стартовой вершины

function bellmanFord(edges, startVertex) {

const distances = {};

const previousVertices = {};

// Инициализация расстояний и предыдущих вершин

distances[startVertex] = 0;

previousVertices[startVertex] = null;

// Релаксация ребер

for (let i = 1; i < Object.keys(edges).length; i++) {

edges.forEach(edge => {

const { source, target, weight } = edge;

if (distances[source] !== undefined && (distances[target] === undefined || distances[source] + weight < distances[target])) {

distances[target] = distances[source] + weight;

previousVertices[target] = source;

}

});

}

// Проверка наличия отрицательных циклов

edges.forEach(edge => {

const { source, target, weight } = edge;

if (distances[source] + weight < distances[target]) {

throw new Error('Граф содержит отрицательные циклы');

}

});

return { distances, previousVertices };

}

Асимптотическая оценка

Асимптотическая оценка алгоритма Форда-Беллмана зависит от числа вершин в графе (|V|) и числа ребер (|E|). Общая сложность алгоритма составляет O (|V| \* |E|), что делает его достаточно эффективным для небольших графов.

Сравнение алгоритма с другими

Алгоритм Форда-Беллмана является одним из классических алгоритмов для решения задачи нахождения кратчайшего пути во взвешенном графе. Однако есть еще алгоритмы, которые выполняют такие же функции, рассмотрим их различия и рассмотрим их асимптотику:

* Алгоритм Дейкстры: Алгоритм Дейкстры также используется для нахождения кратчайшего пути во взвешенном графе, но он применим только к графам с неотрицательными весами ребер и находит расстояние только до заданной вершины. В отличие от алгоритма Форда-Беллмана, алгоритм Дейкстры работает за время , что делает его эффективнее для графов с положительными весами ребер.
* Алгоритм A\*: Алгоритм A\* является эвристическим алгоритмом поиска пути, который комбинирует информацию о стоимости достижения текущей вершины с эвристической оценкой стоимости достижения целевой вершины. Алгоритм A\* обычно применяется в задачах поиска пути в графах, где требуется эффективно находить оптимальные пути. В отличие от алгоритма Форда-Беллмана, алгоритм A\* может быть более эффективным, если имеется эвристическая функция, которая хорошо приближает оценку расстояния до цели.
* Алгоритм Беллмана-Форда с очередью с приоритетами: это оптимизированная версия алгоритма Форда-Беллмана, которая использует очередь с приоритетами для выбора вершин с наименьшей оценкой расстояния. Эта оптимизация позволяет сократить количество повторных просмотров вершин и сделать алгоритм более эффективным.

Сравнение этих алгоритмов зависит от особенностей конкретной задачи и свойств графа. Алгоритм Форда-Беллмана является универсальным и применимым к графам с отрицательными весами ребер, в то время как другие алгоритмы имеют свои ограничения и предположения о графе. Если в графе нет отрицательных циклов, то алгоритм Форда-Беллмана может быть предпочтительным выбором, особенно когда важна общая применимость и гибкость алгоритма. Однако, если граф имеет специфические свойства или ограничения, другие алгоритмы могут быть более эффективными или предоставлять дополнительные возможности, такие как использование эвристической информации или ограничений на веса ребер.

Примеры работы

|  |  |
| --- | --- |
| Исходный граф | Фактический результат |
| **Изображение выглядит как карта, текст, диаграмма, атлас  Автоматически созданное описание** | **Изображение выглядит как текст, снимок экрана, чек, Шрифт  Автоматически созданное описание** |
| **Изображение выглядит как карта, текст, атлас, диаграмма  Автоматически созданное описание** | **Изображение выглядит как текст, снимок экрана, чек, Шрифт  Автоматически созданное описание** |
| **Изображение выглядит как карта, текст, атлас, диаграмма  Автоматически созданное описание** | **Изображение выглядит как текст, снимок экрана, Шрифт, число  Автоматически созданное описание** |
| **Изображение выглядит как карта, текст, атлас, диаграмма  Автоматически созданное описание** | **Изображение выглядит как текст, снимок экрана, Шрифт, чек  Автоматически созданное описание** |

Заключение

В заключение, алгоритм Форда-Беллмана является важным и мощным инструментом для решения задачи нахождения кратчайшего пути во взвешенном графе. Он обладает рядом преимуществ, таких как универсальность, способность работать с графами с отрицательными весами ребер и простота реализации.

В ходе выполнения курсовой работы была проведена тщательная аналитическая и практическая работа по изучению и реализации алгоритма Форда-Беллмана. Были изучены его основные принципы, математическая модель и время работы. Также был проведен анализ и сравнение алгоритма с другими алгоритмами для нахождения кратчайшего пути.

В итоге алгоритм Форда-Беллмана является мощным инструментом, который позволяет эффективно определить кратчайшее время пути груза от заказчика до склада. Его простата и гибкость делают его незаменимым инструментом для решения широкого спектра задач в различных областях.

Список литературы

1. Алгоритм Форда-Беллмана: [Электронный ресурс]. URL: <https://habr.com/ru/companies/otus/articles/484382/> .(Дата обращения: 01.06.2023)
2. Работа с данными и событиями модели мультимаршрута – Технологии Яндекса: [Электронный ресурс]. URL: <https://yandex.ru/dev/maps/jsbox/2.1/multiroute_data_access/> .(Дата обращения: 01.06.2023)
3. Программная работа с маршрутами. Документация Яндекс**:** [Электронный ресурс]. URL: <https://yandex.ru/dev/maps/jsapi/doc/2.1/dg/concepts/router/multiRouter.html#multiRouter__get-active-route> .(Дата обращения: 01.06.2023)
4. Определение координат. Яндекс:[Электронный ресурс]. URL: <https://yandex.ru/map-constructor/location-tool/> .(Дата обращения: 01.06.2023)
5. Алгоритм Форда-Беллмана.Emaxx: [Электронный ресурс]. URL: <https://e-maxx.ru/algo/ford_bellman> .(Дата обращения: 01.06.2023)
6. Алгоритм Форда-Беллмана. Викиконспеты: [Электронный ресурс]. URL: <https://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Форда-Беллмана> .(Дата обращения: 01.06.2023)

Программные приложения

<https://replit.com/@mbatimel/Dostavishok?v=1>