# ЭКСТРЕМАЛЬНЫЕ ЗАДАЧИ НА ГРАФАХ

**1. Задача о кратчайшем пути между двумя вершинами ориентированного графа и ее экономическая интерпретация.**

# Постановка задачи

Задан конечный ориентированный граф $G(V, E)$. Каждой дуге графа “$e$” поставлено в соответствие некоторое число $l(e) \geq 0$, называемое длиной дуги “$e$”. Длиной пути μ называется сумма длин дуг, составляющих данный путь.

$$l(\mu) = \sum_{\mu \epsilon e} l(e) .$$

Требуется для двух фиксированных вершин $v_o$ и $v_n$ графа $G(V, E)$ найти самый короткий соединяющий их путь.

# Экономическое содержание задачи

Задана сеть дорог, соединяющих пункты $v_i$ $(i = 0,1, \dots, n)$. Найти путь, соединяющий пункты $v_o$ и $v_n$, по которому можно доставить груз в кратчайшее время. При этом время доставки груза из пункта $v_i$ в $v_j$ $(i, j \in 0, \dots, n)$ задано и равно $l(e_{ij}) = l(v_i, v_j) \geq 0$.

Если под длиной дуги $l(v_i, v_j)$ понимать стоимость перевозки груза из пункта $v_i$ в $v_j$, то содержание задачи составит нахождение такого пути из пункта $v_0$ в $v_n$, на котором затраты на транспортировку были бы минимальными.

# Алгоритм Форда

Алгоритм решения этой задачи позволяет определить кратчайший путь и его длину за конечное число шагов. Каждая вершина графа получает некоторую числовую метку на первом шаге. Затем метки могут меняться, становясь на некотором шаге постоянным числом. Установившаяся метка данной вершины есть кратчайшее расстояние от этой вершины до вершины $v_0$. Если пути, соединяющего $v_0$ и $v_n$, не существует, будем считать длину кратчайшего пути между этими вершинами равной $+\infty$.

# Шаги алгоритма

**1)** На первом шаге ставим следующие метки:

для вершины $v_0$ $\lambda_0 = 0$, для любой другой вершины $v_i$:

$$\lambda_i = +\infty \ (i = 1, \dots, n).$$

# Алгоритм Форда

**2)** Ищем на графе такую дугу $(v_i, v_j)$, для которой

$$\lambda_j - \lambda_i > l(v_i, v_j).$$

Причем разность $\infty - \infty$ считаем равной 0. Если такая дуга найдется, меняем метку вершины $v_j$ на $\lambda_j = \lambda_i + l(v_i, v_j)$. Если такой дуги не найдется, то пути, соединяющего $v_0$ с $v_n$, не существует.

**3)** Повторяем процедуру пункта 2 до тех пор, пока метки вершин не перестанут меняться.

# Алгоритм Форда

Установившиеся метки обозначим $\lambda^*_i$ $(i = 1,2, \dots, n)$. При этом может быть два случая:

**1)** $\lambda^*_n = +\infty$.

Это значит, что пути, соединяющего $v_0$ и $v_n$, не существует. Длина кратчайшего пути равна $+\infty$.

**2)** $\lambda^*_n$– конечное число. Оно равно кратчайшему расстоянию между вершинами $v_0$ и $v_n$.

# Алгоритм Форда

Кратчайший путь получаем следующим образом. Ищем вершину $v_{p1}$ такую, что $\lambda n - \lambda p1 = l(v_{p1}, v_n)$, затем $v_{p2}$, для которой $\lambda_{p1} - \lambda_{p2} = l(v_{p2}, v_{p1})$ и т.д. до тех пор, пока не придем в вершину $v_{p(k+1)} = v_0$. Путь, проходящий через отмеченные вершины $v_0, v_{pk}, v_{p(k-1)}, \dots, v_{p2}, v_{p1}, v_n$, является кратчайшим.

Как следует из построения и правил изменения меток, метки вершины $v_i$ $(i = 1, \dots, n)$ могут меняться конечное число раз (метка вершины $v_0$ $\lambda_0 = 0$ не меняется), т.к. конечная метка всякой вершины равна длине некоторого пути из $v_0$ в данную вершину $v_i$.

# Алгоритм Форда

Из построения следует, что отмеченный путь $\mu(v_0, v_{pk}, v_{p(k-1)}, \dots, v_{p2}, v_{p1}, v_n)$ – есть кратчайший путь.

Заметим, что решение задачи может не быть однозначным, т.е. существует несколько путей минимальной длины из вершины $v_o$ в $v_n$.

Решение данной задачи можно ускорить, сократив число шагов, если пользоваться формулой $\lambda_j = \min\{\lambda_i + l(v_i, v_j)\}$.

# Сети. Отношение порядка между вершинами ориентированного графа.

- Ориентированный граф без циклов, имеющий одну вершину без входящих дуг (вход графа, источник) и одну вершину без выходящих дуг (выход графа, сток), называется ***сетью***.

Отыскание экстремальных путей на графах такого вида используется в различных экономических расчетах. К их числу относятся рассмотренная выше задача, а также задачи сетевого планирования.

.

# Отношение порядка

В любом ориентированном графе без циклов можно установить ***отношение порядка*** между его вершинами.

Вершина $v_i$ предшествует вершине $v_j$, если существует путь из $v_i$ в $v_j$. Это отношение порядка удовлетворяет аксиомам порядка:

**1)** если $v_i$ предшествует $v_j$, то $v_j$ не предшествует $v_i$;

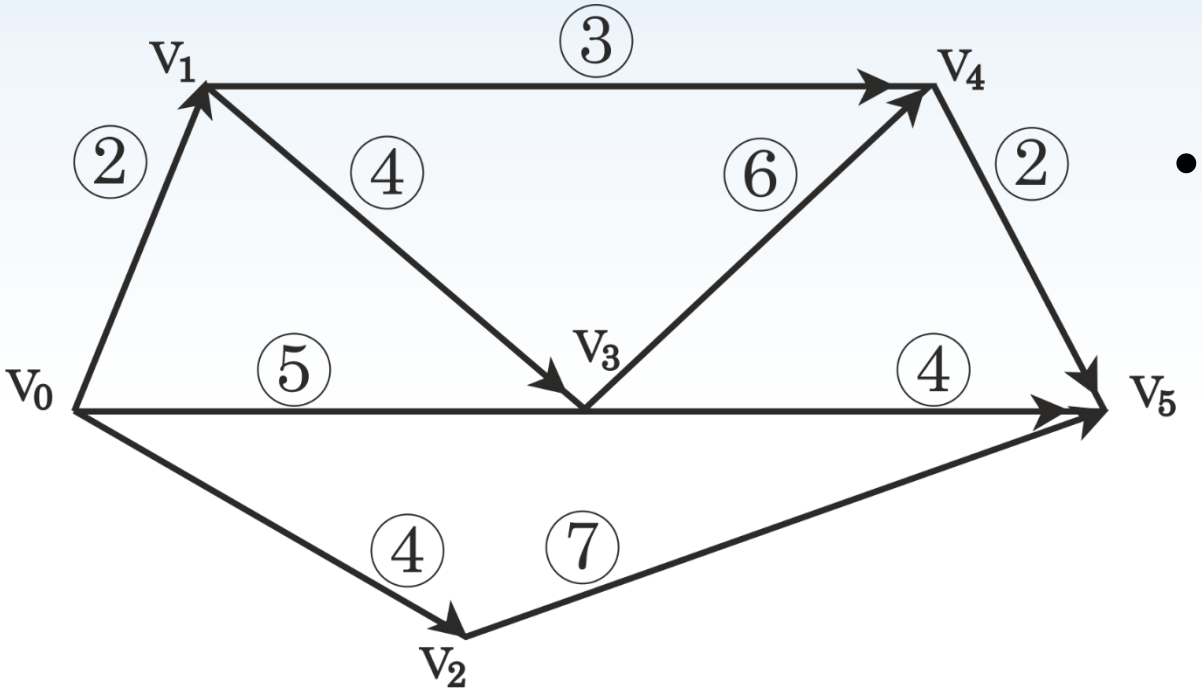
**2)** если $v_i$ предшествует $v_j$, $v_j$ предшествует $v_k$, то $v_i$ предшествует $v_k$.

# Отношение порядка

“Правильная” нумерация вершин графа заключается в том, что при $v_i$ предшествующему $v_j$ номера $i$ и $j$ должны удовлетворять неравенству $i < j$.

На графе-сети практически это можно сделать, используя распределение вершин по рангам методом вычеркивания дуг. Вычеркиваем дуги, исходящие из входа графа, вершины $v_o$. Вершины, соответствующие концам этих дуг и не имеющие после этой операции входящих дуг, относим к вершинам 1-го ранга.

# Пример




На графе $G$ вершинами 1-го ранга являются вершины $v_1$ и $v_2$.

- Вершины 1-го ранга получают первые порядковые номера 1,2. Внутри одного ранга нумерация произвольна.
- Затем вычеркиваем дуги, выходящие из вершин 1-го ранга. Вершины, соответствующие концам таких дуг и не имеющие после этих операций входящих дуг, относим к вершинам 2-го ранга. Они получают следующие порядковые номера. В нашем примере к вершинам 2-го ранга относится одна вершина $v_3$.

# Отношение порядка

Процесс вычеркивания дуг продолжается до тех пор, пока все вершины графа не будут занумерованы. Последний порядковый номер получит вершина $v_n$– выход графа.

В рассмотренном примере все вершины распределены по 4 рангам. К вершинам 3-го ранга принадлежит $v_4$, а к вершинам 4-го ранга – $v_5$.

Существуют и другие способы “правильной” нумерации вершин графа, в том числе алгоритм Форда для нумерации вершин графа.

# 2. Задача о пути максимальной длины между двумя вершинами ориентированного графа в сетевом планировании.

# о пути максимальной длины

- Задача ставится следующим образом. Задан конечный ориентированный граф без контура $G(V, E)$. Каждой дуге графа “$e$” ставится в соответствие длина дуги $l(e)$. Требуется определить длиннейший путь, соединяющий две вершины графа $v_0$ и $v_n$.
- Аналогичные задачи можно ставить для ориентированных графов с контурами, а также неориентированных графов. Но в этом случае во избежание бессодержательности задачи нужно вводить дополнительные условия, исключающие пути бесконечной длины.

# о пути максимальной длины

Решение задачи состоит как в отыскании пути максимальной длины между двумя фиксированными вершинами графа, так и в определении величины этого пути.

Одну из основных задач сетевого планирования составляет отыскание путей максимальной длины между входом и выходом графа-сети.

# Алгоритм

Каждая вершина графа получает числовую метку, которая может меняться конечное число раз. Установившаяся метка – величина длиннейшего пути из вершины $v_0$ в данную вершину $v_j$. В частности, установившаяся метка вершины $v_n$ есть величина длиннейшего пути из $v_0$ в $v_n$.

Чтобы определить искомый путь, нужно рассмотреть последовательность шагов, на каждом из которых ищется одна из дуг длиннейшего пути между $v_0$ в $v_n$.

# Этапы алгоритма

**1)** Полагаем $\lambda_0 = 0;\ \lambda_i = -\infty\ \ (i\ =\ 1, \dots, n)$.

**2)** Ищем дугу $(v_i, v_j)$ такую, что $\lambda_j - \lambda_i \leq l(v_i, v_j)$ . Если такой дуги нет, то не существует пути, соединяющего $v_0$ и $v_n$. Если такая дуга найдется, то изменяем метку $\lambda_j$ на $\lambda'_j\ = \lambda_i + l(v_i, v_j)$.

**3)** Продолжаем процедуру пункта 2 до тех пор, пока метки вершин $v_i$ не перестанут меняться.

# Этапы алгоритма

Установленные метки обозначим $\lambda_i^*$ . При этом могут встретиться два случая:

**1)** $\lambda_n^* = -\infty$, это соответствует тому, что пути, соединяющего вершины $v_0$ и $v_n$, не существует;

**2)** $\lambda_n^*$ - конечное число. Оно равно длине пути максимальной длины из $v_0$ и $v_n$.

Сам путь находим, отмечая вершины, по которым достигается максимум, т.е. те вершины, для которых

$\lambda_j^* = \lambda_i^* + l(v_i, v_j)$.

# Этапы алгоритма

Если между вершинами графа-сети установлено отношение порядка, т.е. они “правильно” занумерованы, то решение задачи можно получить за один шаг, произведя подсчет меток с учетом следующей формулы:

$$\lambda_j = max\{\lambda_i + l(v_i, v_j)\}.$$

# Сетевое планирование. Скорейшее время завершения проекта.

Рассмотрим проект – совокупность операций (работ), составляющий некоторый многошаговый процесс. Примером может служить строительство некоторого объекта. Считаем известными все работы, которые предстоит совершить, их последовательность и время, необходимое для выполнения каждой работы.

Проект может быть изображен в виде графа-сети. Зададимся целью определить кратчайший срок завершения проекта.

# Пусть данные о строительстве приведены в следующей таблице:

| Виды работ | Какие работы следуют за перечисленными | Продолжительность работ |
|---|---|---|
| 1 | 2, 3 | 2 |
| 2 | 8 | 3 |
| 3 | 6, 7 | 4 |
| 4 | 6, 7 | 5 |
| 5 | 9 | 4 |
| 6 | 8 | 6 |
| 7 | - | 4 |
| 8 | - | 2 |
| 9 | - | 7 |

# Сетевое планирование

Эту информацию о проекте представим в виде графа-сети. Дугами графа будем изображать работы, а вершины графа – некоторые события.

Назовем элементарными событиями начало и конец каждой работы, а некоторую совокупность элементарных событий – событием.

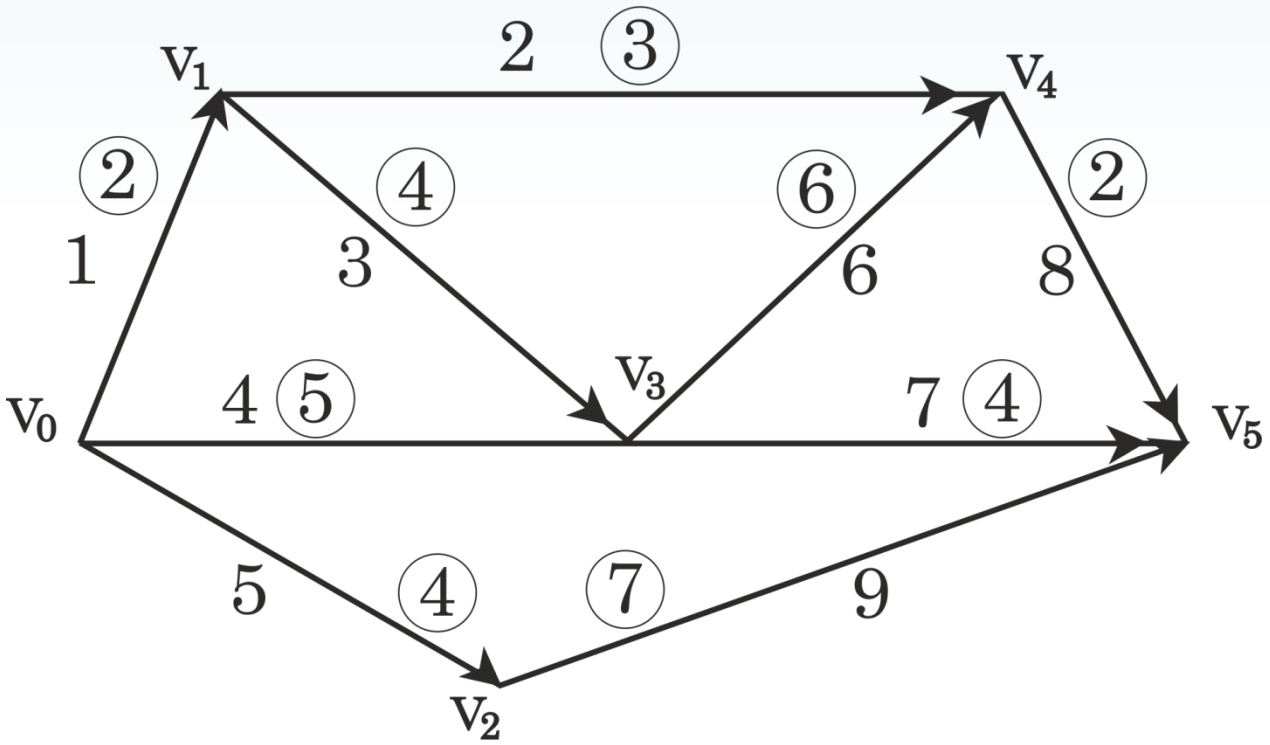
# Сетевое планирование

- Вход графа – событие, заключающееся в начале всего проекта. Оно является событием, стоящим в начале одной или нескольких работ, а именно тех, которые не следуют ни за какими другими, т.е. работ, с которых может быть начато строительство. В нашем примере такими работами являются №1,4,5 (их нет во 2- столбце).
- Выходом графа будет являться событие, заключающееся в окончании работ, за которыми не следуют никакие другие работы, т.е. в окончании всего проекта. В данном примере – это работы №7,8,9.
- Все другие вершины графа есть события, заключающиеся в окончании одних и начале других работ.

# Сетевое планирование

Сетевой граф, соответствующий приведенным в таблице данным, изображен на рисунке.

Номер работы обозначен числом вне кружка. Число, обведенное кружком, есть продолжительность данной работы.



# Сетевое планирование. Пример

- Вход графа, вершина $v_0$ – начало проекта. Выход графа, вершина $v_5$ – окончание проекта.
- Вершины $v_1, v_2, v_3, v_4$ есть события, заключающиеся в начале одних и окончании других работ. Так, например, вершина $v_3$ есть окончание 3-й и 4-й работ и начало 6-й и 7-й.
- Путь максимальной длины из вершины $v_0$ в $v_i$ есть скорейшее время наступления события $v_i$ . В самом деле, событие $v_3$, например, соответствующее началу 6-й и 7-й работ, может произойти только после окончания 3-й и 4-й работ, а следовательно, и после окончания 1-й, т.к. для выполнения 3-й работы необходимо окончание 1-й работы.

# Сетевое планирование

Следовательно, скорейшее время наступления события $v_3$ есть $\max\{5, (2+4)\} = 6$

Скорейшее время наступления события $v_5$ есть скорейшее время окончания проекта в целом и равно длине пути максимальной длины из вершины $v_0$ в $v_5$.

# Сетевое планирование

Итак, если $v_0$ и $v_n$ есть вход и выход графа-сети, соответствующего данному проекту, то для определения наиболее раннего срока окончания всех работ нужно найти путь максимальной длины из $v_0$ в $v_n$, т.е. критический путь, и определить его длину. Время, соответствующее скорейшему окончанию работ, т.е. скорейшему завершению проекта, называется критическим временем данного проекта. Оно численно совпадает с длиной критического пути из $v_0$ в $v_n$.

# Сетевое планирование

В приведенном примере критический путь, проходящий через вершины $v_0, v_1, v_3, v_4, v_5$, имеет длину, равную 14, $l(\mu) = 14$, т.е. критическое время данного проекта равно 14.

Работы, составляющие критический путь, называются критическими работами (операциями). От своевременного выполнения критических операций зависит срок завершения проекта. Они не допускают запаздывания в исполнении в отличие от некритичных операций.