ITIVITI

LET IT TRADE

# Алгоритмы и структуры данных

Косяков Михаил Сергеевич
к.т.н., доцент кафедры ВТ

УНИВЕРСИТЕТ ИТМО

# Содержание курса

- Введение в теорию алгоритмов
- Алгоритмы сортировок
- Структуры данных
  - Линейные структуры
  - Бинарные деревья поиска
  - Хеши и хеш-функции
- **Алгоритмы на графах**
  - Обходы графов в ширину и глубину
  - Минимальные остовные деревья
  - **Поиск кратчайших путей в графе**

ITIVITI
LET IT TRADE
УНИВЕРСИТЕТ ИТМО

ITIVITI

# Элементы динамического программирования

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

# Элементы динамического программирования


"I spent the Fall quarter (of 1950) at RAND. My first task was to find a name for multistage decision processes. An interesting question is, Where did the name, dynamic programming, come from? The 1950s were not good years for mathematical research. We had a very interesting gentleman in Washington named Wilson. He was Secretary of Defense, and he actually had a pathological fear and hatred of the word, research. I'm not using the term lightly; I'm using it precisely. His face would suffuse, he would turn red, and he would get violent if people used the term, research, in his presence. You can imagine how he felt, then, about the term, mathematical. The RAND Corporation was employed by the Air Force, and the Air Force had Wilson as its boss, essentially. Hence, I felt I had to do something to shield Wilson and the Air Force from the fact that I was really doing mathematics inside the RAND Corporation. What title, what name, could I choose? In the first place I was interested in planning, in decision making, in thinking. But planning, is not a good word for various reasons. I decided therefore to use the word, "programming" I wanted to get across the idea that this was dynamic, this was multistage, this was time-varying I thought, lets kill two birds with one stone. Lets take a word that has an absolutely precise meaning, namely dynamic, in the classical physical sense. It also has a very interesting property as an adjective, and that is its impossible to use the word, dynamic, in a pejorative sense. Try thinking of some combination that will possibly give it a pejorative meaning. Its impossible. Thus, I thought dynamic programming was a good name. It was something not even a Congressman could object to. So I used it as an umbrella for my activities."

Richard Bellman, *Eye of the Hurricane: an autobiography*, 1984.

LET IT TRADE

# Элементы динамического программирования

ITIVITI

- Числа Фибоначчи
  - $F_0 = 0, F_1 = 1$
  - $F_n = F_{n-1} + F_{n-2}, n \geq 2$

```
01: int fibonacci(int n)
02: {
03:     if (n <= 1)
04:         return n;
05:     return fibonacci(n-1) + fibonacci(n-2);
06: }
```

- Какое дерево рекурсии?

LET IT TRADE



УНИВЕРСИТЕТ ИТМО

# Элементы динамического программирования

ITIVITI

- Числа Фибоначчи. Метод 1
  - Нисходящий с запоминанием (memoization)

```
01: int calculated[MAX];
02:
03: void initialize() {
04:   int i;
05:   for (i = 0; i < MAX; ++i)
06:     calculated[i] = -1;
07: }
08:
09: int fibonacci(int n) {
10:    if(calculated[n] == -1) {
11:     if (n <= 1)
12:       calculated[n] = n;
13:     else
14:       calculated[n] = fibonacci(n-1) + fibonacci(n-2);
15:    }
16:    return calculated[n];
17: }
```

LET IT TRADE



УНИВЕРСИТЕТ ИТМО

# Элементы динамического программирования

- Числа Фибоначчи. Метод 2
  - Восходящий табличный (tabular)

```
01: int fibonacci(int n)
02: {
03:   int calculated[n];
04:   int i;
05:   calculated[0] = 0;   calculated[1] = 1;
06:   for (i = 2; i <= n; ++i)
07:       calculated[i] = calculated[i-1] + calculated[i-2];
08:
09:   return calculated[n];
10: }
```

# Элементы динамического программирования

ITIVITI

- Задача про кузнечика
  - Сколько способов допрыгать до $n$ ?
  - Наибольшее число монет, которое мы можем получить, допрыгав до $n$ ?
  - Как восстановить путь?
- Оптимальное решение задачи выражается через оптимальные решения подзадач
  - Строим оптимальное решение задачи из оптимальных решений подзадач
- Подзадачи перекрываются
  - Решаем подзадачу один раз и запоминаем решение

LET IT TRADE

ITIVITI

# Поиск кратчайших путей в графе (динамическое программирование)

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

# Кратчайшие пути из одной вершины

ITIVITI

- Вспоминаем алгоритм Дейкстры
  - Плюсы: для связного графа время работы $O(m \log n)$ на бинарной пирамиде
  - Минусы: не подходит, если есть ребра с отрицательным весом; централизованный
- Может ли в кратчайший путь входить цикл с отрицательным весом?
  - Необходимо выявить наличие и сообщить об этом
  - Пока предположим, что таких циклов нет
- Может ли в кратчайший путь входить цикл с положительным весом?

LET IT TRADE

# Структура кратчайшего пути

- Пусть граф $G$ не имеет отрицательных циклов. Сколько ребер включает в себя кратчайший путь из вершины $s$ в произвольную вершину $v$?
  - Произвольное число ребер
  - Число ребер в пути $\leq m$
  - Число ребер в пути $\leq n - 1$
  - Число ребер в пути $\leq n$

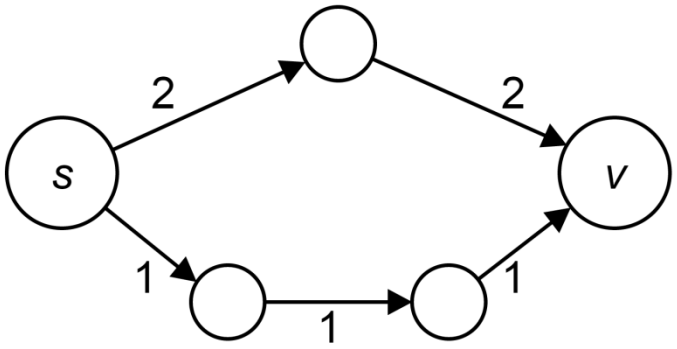
LET IT TRADE



ITIVITI
УНИВЕРСИТЕТ ИТМО

# Структура кратчайшего пути

- Подпути кратчайшего пути сами д.б. кратчайшими
- Как можно выразить оптимальное решение через оптимальные решения подзадач?
  - Что есть подзадача более крупной задачи?
  - Основная идея: давайте ограничим число ребер в пути
  - Размер подзадачи эквивалентен числу разрешенных ребер в пути






LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

# Структура кратчайшего пути

- **Основная лемма:** Пусть $P$ – кратчайший путь в графе $G$ (возможно с отрицательными циклами) из вершины $s$ в произвольную вершину $v$, состоящий из не более чем $i$ ребер
  - **Случай 1**: если $P$ содержит не более $(i-1)$ ребра, то $P$ является кратчайшим путем из $s$ в $v$, состоящим из не более чем $(i-1)$ ребра
  - **Случай 2:** если $P$ содержит ровно $i$ ребер с последним ребром $(u, v)$, то подпуть $P'$ из $s$ в $u$ является кратчайшим путем из $s$ в $u$, состоящим из не более чем $(i-1)$ ребра

# Структура кратчайшего пути

- Из какого числа оптимальных решений подзадач (кратчайших путей с числом ребер $\le i-1$) строится оптимальное решение задачи (кратчайший из $s$ в $v$ путь с числом ребер $\le i$) ?
  - $n-1$
  - $n$
  - $1 + \deg^+(v)$
  - 2

# Структура кратчайшего пути

- Пусть $W_{i,v}$ – минимальный вес пути в графе $G$ (возможно с отрицательными циклами) из $s$ в $v$ с числом ребер $\le i$ ($W_{i,v} = +\infty$, если такого пути нет)
- тогда для произвольной вершины $v$ и $i \ge 1$

$$W_{i,v} = \min \begin{cases} W_{i-1,v} \\ \min_{(u,\,v)} \{W_{i-1,u} + weight\,(u,\,v)\} \end{cases}$$

- Если отрицательных циклов нет, то достаточно рассчитывать $W_{i,v}$ до значений $i = n - 1$

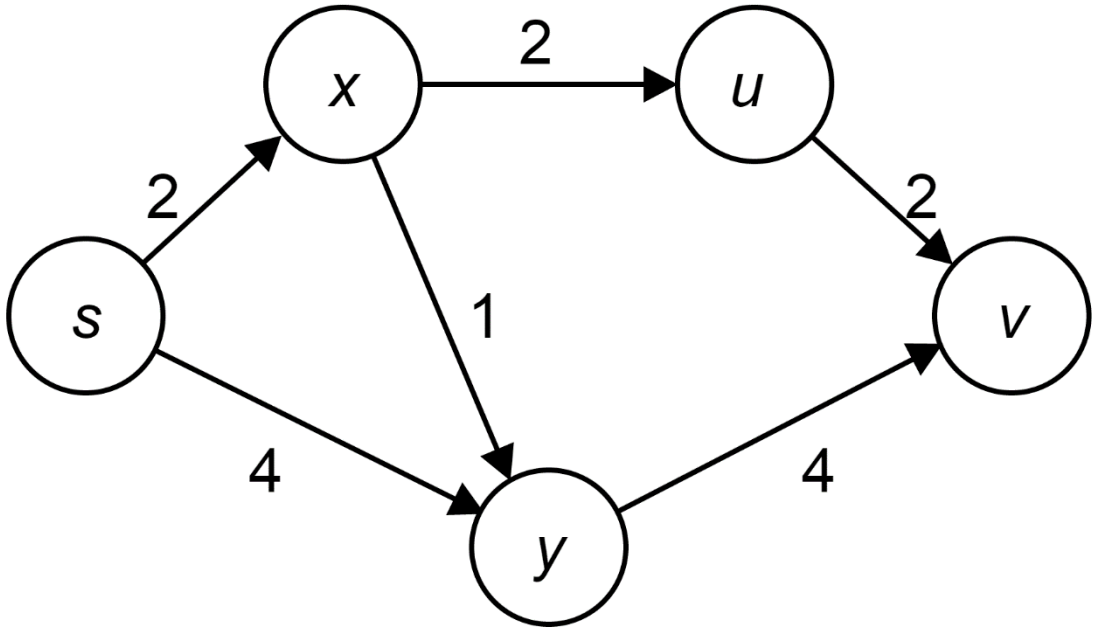
# Алгоритм Беллмана-Форда: реализация


```
01: for each v in V:
02: if v != s:
03:     W[0][v] = ∞  # кратчайший путь, содержащий ноль ребер
04: else:
05:     W[0][v] = 0 # кратчайший путь, содержащий ноль ребер
06:
07: for i=1 to n-1:
08:     for each v in V:
09:         W[i][v] = W[i-1][v]
10:     for each edge (u,v) in E:
11:         if W[i][v] > W[i-1][u] + weight(u,v):
12:            W[i][v] = W[i-1][u] + weight(u,v)
```

- Если в $G$ нет отрицательных циклов, то алгоритм корректен
  - Для вычисления $W_{i,v}$ делаем полный перебор из всевозможных кандидатов (подзадач)

LET IT TRADE

# Алгоритм Беллмана-Форда: пример выполнения




- Ранний останов: можем ли мы остановиться при $i < n - 1$ ?
- Какое время выполнения?



ITIVITI
LET IT TRADE
УНИВЕРСИТЕТ ИТМО

# Алгоритм Беллмана-Форда: обнаружение отриц. циклов


LET IT TRADE

- Что если в $G$ есть отрицательные циклы?
- Для любой вершины $v$ на цикле $W_{i,v} \rightarrow -\infty$ (*)
  - Насколько большое $i$ нам нужно?
- **Лемма:** в графе $G$ отсутствуют отрицательные циклы, достижимые из $s$, тогда и только тогда, когда $W_{n-1,v} = W_{n,v}$ для всех вершин $v$
  - $\Rightarrow$ : уже доказали корректность алгоритма Б-Ф
  - $\Leftarrow$ : $W_{n-1,v} = W_{n,v} \Rightarrow W_{n-1,v} = W_{i,v}$ для всех $i \geq n$ : противоречие с (*)
- Т.о. достаточно всего еще одной итерации Б-Ф!

# Алгоритм Беллмана-Форда: затраты памяти

ITIVITI

LET IT TRADE

- Каковы затраты памяти базового алгоритма Беллмана-Форда?
  - $\Theta(m\,n)$
  - $\Theta(m^2)$
  - $\Theta(n^3)$
  - $\Theta(n^2)$
- Но для вычисления $W_{i,v}$ нам требуется лишь $W_{i-1,v}$ !
- Т.е. $O(1)$ для каждой вершины-назначения!



УНИВЕРСИТЕТ ИТМО

# Алгоритм Беллмана-Форда: затраты памяти

ITIVITI

LET IT TRADE

- Можем сделать еще лучше!

$$W_v = \min [\, W_v \,,\, \min_{(u,\, v)} \{ W_u + weight\,(u,\, v) \} \,]$$

- Через $i$ итераций путь из $s$ в $v$ с весом $W_v$ может содержать значительно больше ребер, чем $i$
  - Как получить вес кратчайшего пути за одну итерацию, если граф – простой путь?
- Через $i$ итераций величина $W_v$ не превосходит веса кратчайшего пути из $s$ в $v$ с числом ребер $\leq i$
- Каковы затраты памяти в этом случае?



УНИВЕРСИТЕТ ИТМО

# Алгоритм Беллмана-Форда: реализация


```
01: for each v in V:
02:     v.d = ∞  # кратчайший путь, содержащий ноль ребер
03:     v.p = NIL
04: s.d = 0 # кратчайший путь, содержащий ноль ребер из s в s
05: for i=1 to n-1:
06:     for each edge (u,v) in E:
07:         if v.d > u.d + weight(u,v):
08:             v.d = u.d + weight(u,v)
09:             v.p = u
10: # обнаружение отрицательного цикла
11: for each edge (u,v) in E:
12:     if v.d > u.d + weight(u,v):
13:         return false
14: return true
```

LET IT TRADE



# Алгоритм Беллмана-Форда: восстановление отриц. цикла

ITIVITI

LET IT TRADE

- Рассмотрим вершину $v$, для которой $W_{n-1,v} \neq W_{n,v}$
- Путь $P$ из $s$ в $v$ содержит ровно $n$ ребер
  - значит содержит цикл
- и любой путь из $\leq n-1$ ребер имеет больший вес
  - значит цикл отрицательный: иначе удалим цикл и получим путь из $\leq n-1$ ребер меньшего веса



УНИВЕРСИТЕТ ИТМО

ITIVITI

LET IT TRADE

# Спасибо за внимание!



УНИВЕРСИТЕТ ИТМО