Семинар 14

Динамическое программирование (продолжение)

Задача о нахождении расстояния редактирования

Текстовый редактор, подозревая опечатку, предлагает заменить написанное слово на *близкое (?).*

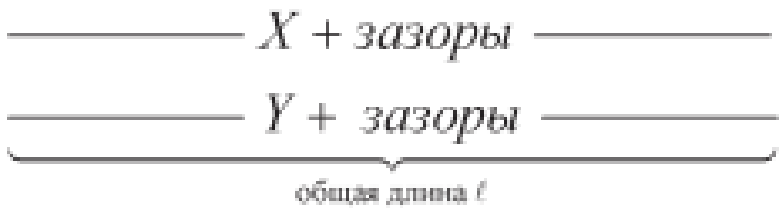
Каково расстояние между словами SNOWY и SUNNY?

S	_	N	O	W	Y	
S	U	N	N	_	Y	

или

_	S	N	O	W	_	Y
S	U	N	_	_	N	Y

Стоимость выравнивания: количество столбцов, в которых символы различаются. Расстояние редактирования между словами – стоимость их наилучшего выравнивания.



- пусть A - конечный алфавит. $X = x_1 x_2 \dots x_m$, $Y = y_1 y_2 \dots y_n$,

$x_i, y_j \epsilon A$. Сколько существует возможностей для содержимого конечного столбца оптимального выравнивания?

а) 2

б) 3

в) 4

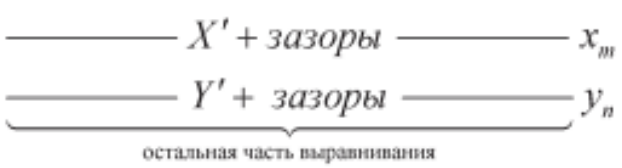
г) mn

3 варианта

1)	2)	3)
x_m	x_m	–
y_n	–	y_n

Вывод рекуррентного соотношение

1) x_m y_n



$P = P' + \alpha$, P'- оптимальное выравнивание для X' и Y'

$$\alpha = \begin{cases} 0, x_m = y_n \\ 1, x_m \neq y_n \end{cases}$$

2) x_m –

3) – y_n

Если одна из цепочек пуста (например, Y), то расстояние редактирования равно X

. *Оптимальное выравнивание двух непустых символьных цепочек* $X = x_1, x_2, \ldots, x_m$ *и* $Y = y_1, y_2, \ldots, y_n$ *равно либо*:

(i) *оптимальному выравниванию* X' *и* Y', *дополненному сочетанием* x_m *и* y_n *в последнем столбце*;

(ii) *оптимальному выравниванию* X' *и* Y, *дополненному сочетанием* x_m *и зазора в конечном столбце*;

(iii) *оптимальному выравниванию* X *и* Y', *дополненному сочетанием зазора и* y_n *в конечном столбце*,

где X' *и* Y' *обозначают соответственно* X *и* Y *с удаленными конечными символами* x_m *и* y_n.

Рекуррентное соотношение

- $P_{i,j} = \min\{P_{i-1,j-1} + \alpha, P_{i-1,j} + 1, P_{i,j-1} + 1\}, \ i = 1,2, \dots, m, j = 1,2, \dots, n$
- Подзадачи: вычислить минимальное расстояние редактирования первых i символов *X* и первых j символов *Y*.

Dist_Red

Вход: цепочки $X = x_1 x_2 \dots x_m, Y = y_1 y_2 \dots y_n$ над алфавитом *A*

Выход: *P* – расстояние редактирования

//решения подзадач(индексируемых с 0)

$A := (m+1)(n+1)$ двумерный массив

// базовый случай 1 $(j = 0)$
$For\ i := 0\ to\ m\ do\ A[i][0] = i$

// базовый случай 2 $(i = 0)$
$For\ j := 0\ to\ n\ do$
$A[0][j] = j$

$O(m \cdot n)$

//систематическое решение всех подзадач
$For\ i := 0\ to\ m\ do$
$For\ j := 0\ to\ n\ do$
$A[i][j] := \min\{A[i-1][j-1] + \alpha, A[i-1][j] + 1, A[i][j-1] + 1\}$

Return $A[m][n]$ //решение самой крупной подзадачи

Пример вычисления расстояния редактирования

Вычислить расстояние редактирования между словами «МУСОР» и «ССОРА»

а	5	5	5	5	4	3
р	4	4	4	4	3	2
о	3	3	3	3	2	3
с	2	2	2	2	3	4
с	1	1	2	2	3	4
	0	1	2	3	4	5
		м	у	с	о	р

Алгоритм реконструкции (обратный проход)

Зазор в слове «МУСОР»

Зазор в слове «ССОРА»

Сверху и снизу нет зазора

–	С	С	О	Р	А	
М	У	С	О	Р	–	

Выравнивание последовательностей

Сравниваем 2 участка одного или нескольких геномов.

Алфавит A={A, C, G, T} A: Аденин; G — Г: Гуанин; C — Ц: Цитозин;T — Т: Тимин

Допустим, что за несовпадение символов штраф=2, за отсутствие (зазор) штраф=1

Сравним ACG и AGCT

G	3	2	1	2	3
C	2	1	2	1	2
A	1	0	1	2	3
	0	1	2	3	4
		A	G	C	T

A G C T
A – C G

или

A G C – T
A – C G –

или

A G C T –
A – C – G

Кратчайшие пути в ориентированном графе с отрицательными длинами ребер (повторно)

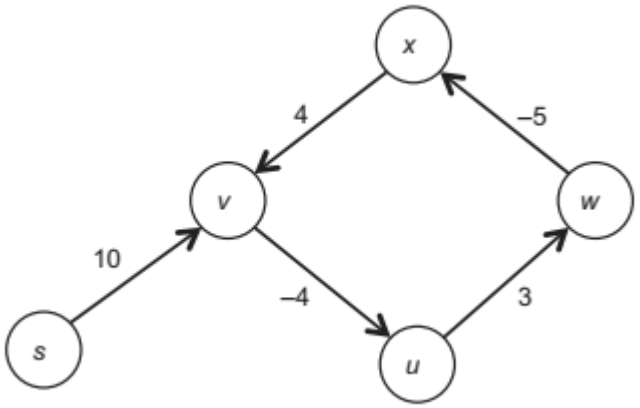
- Задача: кратчайшие пути с единственным истоком

Вход: ориентированный граф $G = (V, E)$, истоковая вершина $s \in V$ и вещественная длина ℓ_e для каждого ребра $e \in E$.

Выход: один из следующих:

(i) расстояние кратчайшего пути $dist(s, v)$ для каждой вершины $v \in V$ либо

(ii) заявление о том, что G содержит отрицательный цикл.



Рассмотрим экземпляр задачи о кратчайшем пути с единственным истоком с n вершинами, m ребрами, стартовой вершиной s и без отрицательных циклов. Что из перечисленного является истинным? Выберите самое подходящее утверждение:

а) Для каждой вершины v, достижимой из истока s, существует кратчайший путь s–v не более чем с $n - 1$ ребрами;

б) Для каждой вершины v, достижимой из истока s, существует кратчайший путь s–v не более чем с m ребрами;

г) Нет конечной верхней границы (как функции от n и m) на наименьшем числе ребер в кратчайшем пути s–v.

Алгоритм Беллмана-Форда с использованием стратегии динамического программирования

- Если задача решена и P – кратчайший путь $s \to v$, содержащий не более, чем i ребер, то

$$L_{i,v} = \min \begin{cases} L_{i-1,v} \\ \min_{(w,v)\in E} \{L_{i-1,w} + \ell_{wv}\} \end{cases} \quad *$$

Алгоритм Беллмана-Форда

BELLMAN—FORD

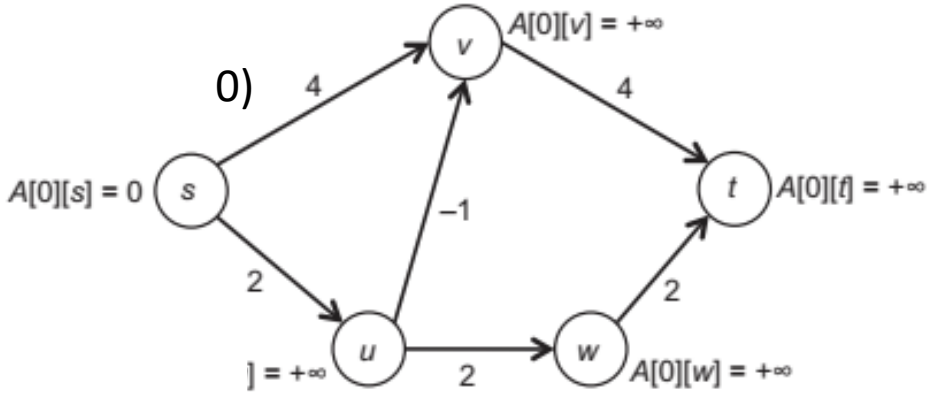
Вход: ориентированный граф $G = (V, E)$, представленный в виде списков смежности, истоковая вершина $s \in V$ и вещественная длина ℓ_e для каждого $e \in E$.

Выход: $dist(s, v)$ для каждой вершины $v \in V$ либо объявление, что G содержит отрицательный цикл.

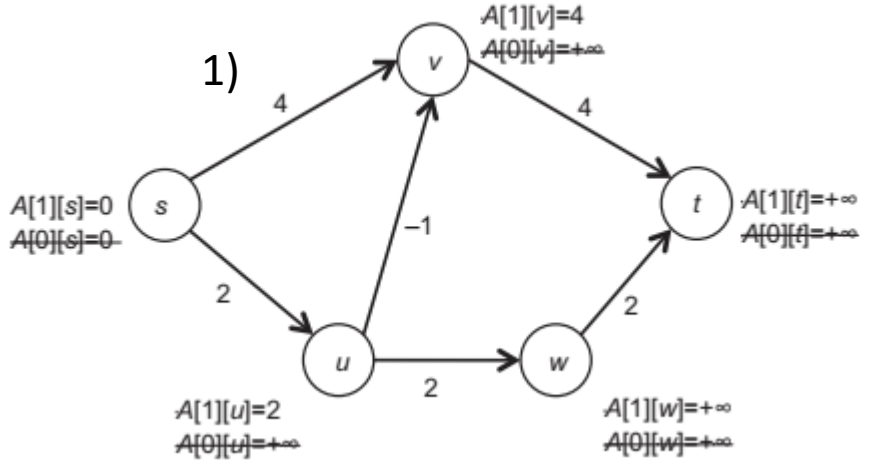
```
// подзадачи (i индексируется от 0, v индексирует V)
A := (n + 1) × n двумерный массив
// базовые случаи (i = 0)
A[0][s] := 0
for each v ≠ s do
    A[0][v] := +∞
// систематически решить все подзадачи
for i = 1 to n do                    // размер подзадачи
    stable := TRUE                   // для досрочной остановки
    for v ∈ V do
        // использовать рекурренцию  *
        A[i][v] :=

        min{ A[i−1][v] (Случай 1), min_{(w,v)∈E} {A[i−1][w] + ℓ_wv} (Случай 2) }

        if A[i][v] ≠ A[i − 1][v] then
            stable := FALSE
    if stable = TRUE then            // выполнено леммой 18.3
        return {A[i − 1][v]}_{v∈V}
// не удалось стабилизироваться на n итерациях
return «отрицательный цикл» // исправлено леммой 18.4
```

Пример

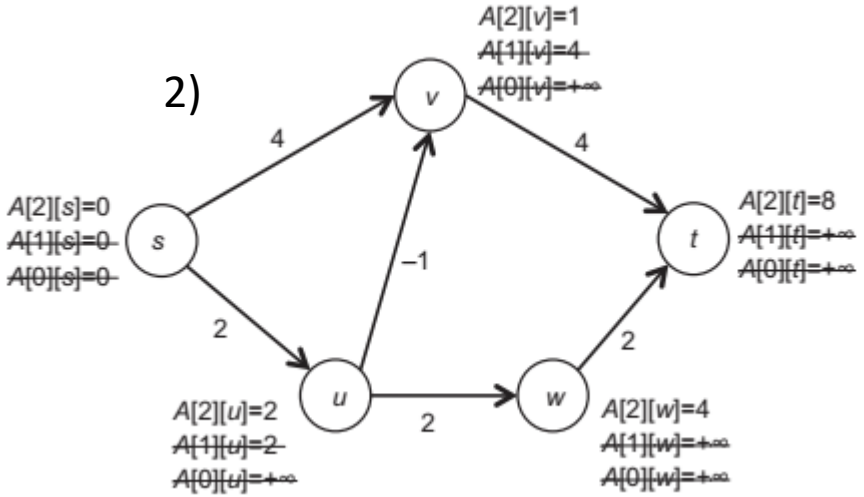
0)



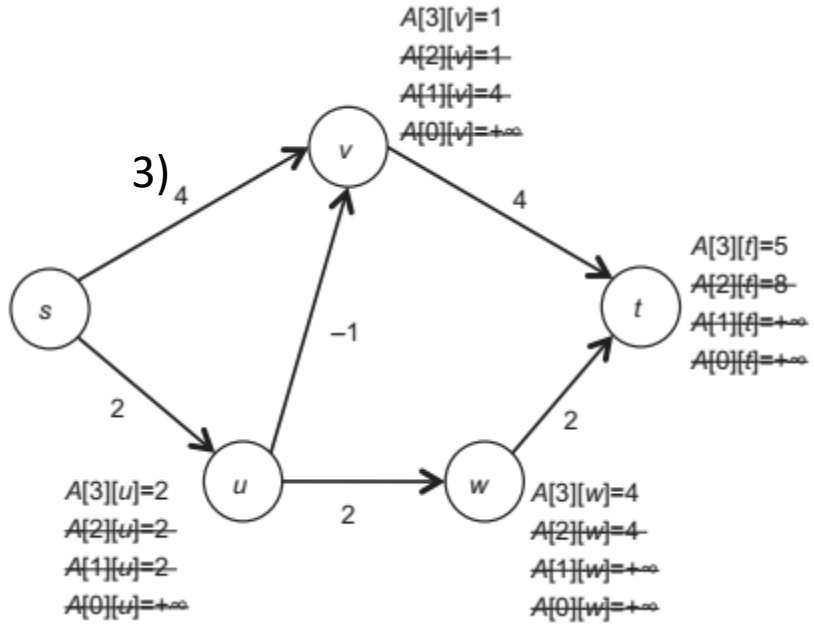
1)



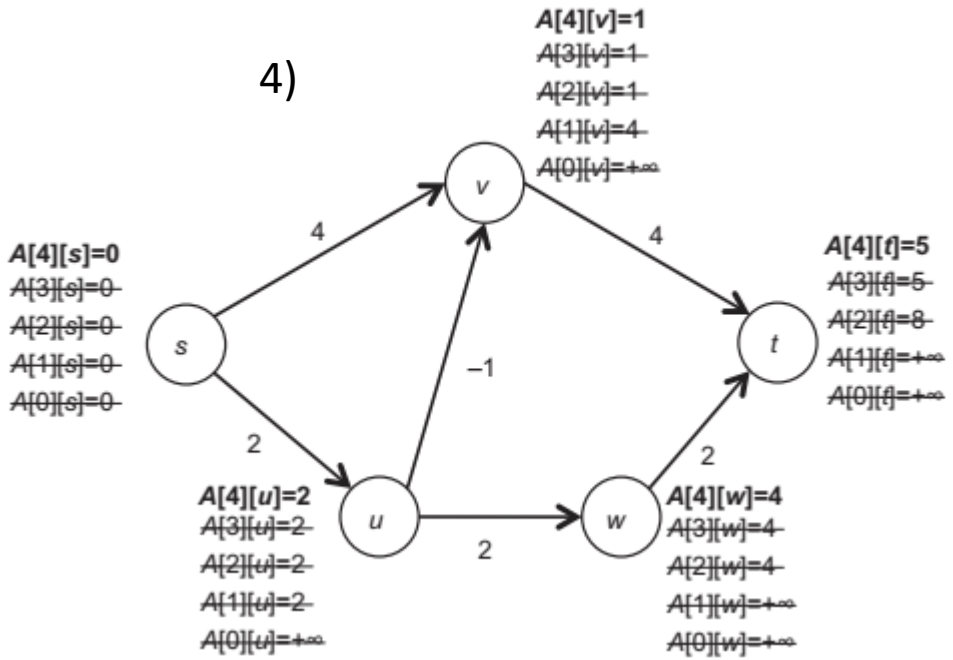
2)

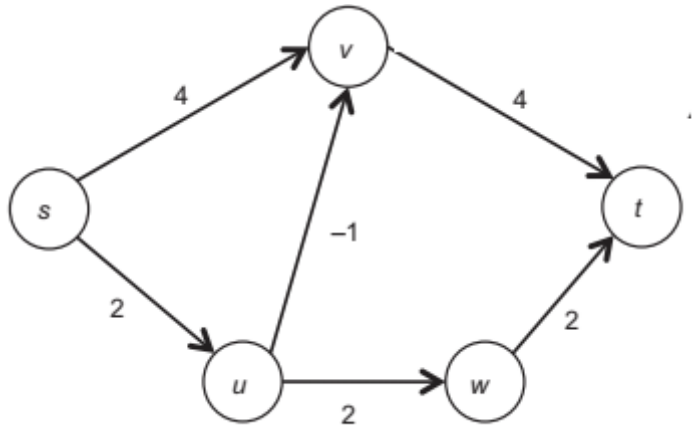


3)



4)





v \ i	0	1	2	3	4
s	0	0	0	0	0
v	∞	4	1	1	1
u	∞	2	2	2	2
w	∞	∞	4	4	4
t	∞	∞	8	5	5

Не меняется

$O(mn)$