Динамическое программирование, третья лекция

Иван Казменко

Кружок обучения мастерству программирования при мат-мехе СПбГУ

Четверг, 20 октября 2011 года

Оглавление

Оглавление

Число сочетаний

Постановка задачи:

- Есть n различных объектов.
- Выбираем из них k различных объектов (порядок не важен).
- Сколько способов это сделать? Обозначим это число как C_n^k.

Объекты можно просто пронумеровать числами от 1 до n.

Пример: $n = 4$, $k = 2$. Получаем $C_4^2 = 6$.

1	2	3	4	Сочетание
1	2	3	4	1, 2
1	2	3	4	1, 3
1	2	3	4	1, 4
1	2	3	4	2, 3
1	2	3	4	2, 4
1	2	3	4	3, 4

Число сочетаний

Постановка задачи:

- Есть n различных объектов.
- Выбираем из них k различных объектов (порядок не важен).
- Сколько способов это сделать? Обозначим это число как C_n^k.

Объекты можно просто пронумеровать числами от 1 до n.
Пример: $n = 4$, $k = 2$. Получаем $C_4^2 = 6$.

1	2	3	4	Сочетание
1	2	3	4	1, 2
1	2	3	4	1, 3
1	2	3	4	1, 4
1	2	3	4	2, 3
1	2	3	4	2, 4
1	2	3	4	3, 4

Число сочетаний

Вычисление C_n^k — комбинаторное решение:

- Выбираем первый объект n способами.
- Выбираем второй объект из оставшихся $n-1$ способом.
- $\ldots$
- Выбираем k-й объект из оставшихся $n-k+1$ способом.
- Каждое сочетание получилось $k!$ раз в различных порядках.
- Поэтому поделим ответ на $k! = 1 \cdot 2 \cdot 3 \cdots k$.

Получаем, что

$$\begin{aligned} C_n^k &= \frac{n \cdot (n-1) \cdots (n-k+1)}{1 \cdot 2 \cdots k} \\ &= \frac{1 \cdot 2 \cdots (n-k)}{1 \cdot 2 \cdots (n-k)} \times \frac{(n-k+1) \cdot (n-k+2) \cdots n}{1 \cdot 2 \cdots k} \\ &= \frac{n!}{k! \times (n-k)!}. \end{aligned}$$

Число сочетаний

Вычисление C_n^k — комбинаторное решение:

- Выбираем первый объект n способами.
- Выбираем второй объект из оставшихся $n-1$ способом.
- ...
- Выбираем k-й объект из оставшихся $n-k+1$ способом.
- Каждое сочетание получилось $k!$ раз в различных порядках.
- Поэтому поделим ответ на $k! = 1 \cdot 2 \cdot 3 \cdots k$.

Итоговая формула:

$$C_n^k = \frac{n!}{k! \times (n-k)!}$$

Пример: $n = 4$, $k = 2$. Получаем

$$C_4^2 = \frac{1 \cdot 2 \cdot 3 \cdot 4}{1 \cdot 2 \times 1 \cdot 2} = \frac{24}{4} = 6.$$

Число сочетаний

Вычисление C_n^k — сведе́ние к подзадачам:

- Рассмотрим объект с номером n.
- Если мы выберем его, из $n-1$ оставшегося объекта нужно будет выбрать ещё $k-1$.
- Если мы не выберем его, из $n-1$ оставшегося объекта нужно будет выбрать ещё k.

Получаем, что

$$C_n^k = C_{n-1}^{k-1} + C_{n-1}^k.$$

База: $C_0^0 = 1$, $C_n^k = 0$ при $k < 0$ или $k > n$.
Пример: $n = 4$, $k = 2$. Получаем $C_4^2 = 6$.

Число сочетаний

n \ k	-1	0	1	2	3	4
0	0	1	0	0	0	0
1	0	?	?	0	0	0
2	0	?	?	?	0	0
3	0	?	?	?	?	0
4	0	?	?	?	?	?

Получаем, что

$$C_n^k = C_{n-1}^{k-1} + C_{n-1}^k.$$

База: $C_0^0 = 1$, $C_n^k = 0$ при $k < 0$ или $k > n$.
Пример: $n = 4$, $k = 2$. Получаем $C_4^2 = 6$.

Число сочетаний

n \ k	-1	0	1	2	3	4
0	0	1	0	0	0	0
1	0	1	1	0	0	0
2	0	1	2	1	0	0
3	0	1	3	3	1	0
4	0	1	4	6	4	1

Получаем, что

$$C_n^k = C_{n-1}^{k-1} + C_{n-1}^k.$$

База: $C_0^0 = 1$, $C_n^k = 0$ при $k < 0$ или $k > n$.
Пример: $n = 4$, $k = 2$. Получаем $C_4^2 = 6$.

Число сочетаний

n \ k	-1	0	1	2	3	4
0	0	1	0	0	0	0
1	0	1	1	0	0	0
2	0	1	2	1	0	0
3	0	1	3	3	1	0
4	0	1	4	6	4	1

Получаем, что

$$C_n^k = C_{n-1}^{k-1} + C_{n-1}^k.$$

База: $C_0^0 = 1$, $C_n^k = 0$ при $k < 0$ или $k > n$.
Пример: $n = 4$, $k = 2$. Получаем $C_4^2 = 6$.

Число сочетаний

n \ k	−1	0	1	2	3	4
0	0	1	0	0	0	0
1	0	1	1	0	0	0
2	0	1	2	1	0	0
3	0	1	3	3	1	0
4	0	1	4	6	4	1

```
for n := 0 upto MAXN:
  c[n][0] := 1
  for k := 1 upto n:
    c[n][k] := c[n - 1][k - 1] + c[n - 1][k]
```

Оглавление

Наибольшая общая подпоследовательность

Определения:

- Подпоследовательность последовательности $p_1, p_2, \ldots, p_k$ (будем обозначать как $p_{1..k}$) — это последовательность $p_{i_1}, p_{i_2}, \ldots, p_{i_r}$ для индексов $1 \leqslant i_1 < i_2 < \ldots < i_r \leqslant k$.
- Общая подпоследовательность $a_{1..m}$ и $b_{1..n}$ — это последовательность, являющаяся подпоследовательностью и для a, и для b.

Постановка задачи:

- Есть две последовательности $a_{1..m}$ и $b_{1..n}$.
- Требуется найти общую подпоследовательность максимальной длины.

Природа объектов, из которых состоят последовательности, здесь не важна.

Поэтому мы будем рассматривать последовательности целых чисел.

Наибольшая общая подпоследовательность

Пример:

- $a = (1,\ 2,\ 4,\ 6,\ 3)$, $m = 5$
- $b = (5,\ 1,\ 4,\ 2,\ 6,\ 3,\ 2)$, $n = 7$

Наибольшая общая подпоследовательность

Пример:

- $a = (1, 2, 4, 6, 3)$, $m = 5$
- $b = (5, 1, 4, 2, 6, 3, 2)$, $n = 7$
- Наибольшая общая подпоследовательность:
  $p = (1, 2, 6, 3)$, её длина равна 4.
  Как подпоследовательность a: $p = (a_1, a_2, a_4, a_5)$.
  Как подпоследовательность b: $p = (b_2, b_4, b_5, b_6)$.

Наибольшая общая подпоследовательность

Пример:

- $a = (1, 2, 4, 6, 3)$, $m = 5$
- $b = (5, 1, 4, 2, 6, 3, 2)$, $n = 7$
- Наибольшая общая подпоследовательность:
  $p = (1, 2, 6, 3)$, её длина равна 4.
  Как подпоследовательность a: $p = (a_1, a_2, a_4, a_5)$.
  Как подпоследовательность b: $p = (b_2, b_4, b_5, b_6)$.
- Другой оптимальный ответ:
  $q = (1, 4, 6, 3)$, её длина также равна 4.
  Как подпоследовательность a: $q = (a_1, a_3, a_4, a_5)$.
  Как подпоследовательность b: $q = (b_2, b_3, b_5, b_6)$.

Наибольшая общая подпоследовательность

Сводим к следующим подзадачам:

- Даны префиксы (начала) последовательностей a и b: $a_{1..i}$ и $b_{1..j}$.
- Найдём их наибольшую общую подпоследовательность.

Решение подзадачи для $a_{1..i}$ и $b_{1..j}$:

- Пусть оптимальное решение непусто (иначе всё очевидно).
- Рассмотрим последнее число в нём.
- Это число равно какому-то a_x $(1 \leqslant x \leqslant i)$.
- Это число также равно какому-то b_y $(1 \leqslant y \leqslant j)$.
- Значит, для получения оптимального решения нужно решить подзадачу для префиксов $a_{1..x-1}$ и $b_{1..y-1}$ и приписать к полученному решению это число (оно равно a_x и b_y).
- Если пар (x, y) несколько, ...

Наибольшая общая подпоследовательность

Сводим к следующим подзадачам:

- Даны префиксы (начала) последовательностей a и b: $a_{1..i}$ и $b_{1..j}$.
- Найдём их наибольшую общую подпоследовательность.

Решение подзадачи для $a_{1..i}$ и $b_{1..j}$:

- Пусть оптимальное решение непусто (иначе всё очевидно).
- Рассмотрим последнее число в нём.
- Это число равно какому-то a_x $(1 \leqslant x \leqslant i)$.
- Это число также равно какому-то b_y $(1 \leqslant y \leqslant j)$.
- Значит, для получения оптимального решения нужно решить подзадачу для префиксов $a_{1..x-1}$ и $b_{1..y-1}$ и приписать к полученному решению это число (оно равно a_x и b_y).
- Если пар (x, y) несколько, ...

Наибольшая общая подпоследовательность

Сводим к следующим подзадачам:

- Даны префиксы (начала) последовательностей a и b: $a_{1..i}$ и $b_{1..j}$.
- Найдём их наибольшую общую подпоследовательность.

Решение подзадачи для $a_{1..i}$ и $b_{1..j}$:

- Пусть оптимальное решение непусто (иначе всё очевидно).
- Рассмотрим последнее число в нём.
- Это число равно какому-то a_x $(1 \leqslant x \leqslant i)$.
- Это число также равно какому-то b_y $(1 \leqslant y \leqslant j)$.
- Значит, для получения оптимального решения нужно решить подзадачу для префиксов $a_{1..x-1}$ и $b_{1..y-1}$ и приписать к полученному решению это число (оно равно a_x и b_y).
- Если пар (x, y) несколько, найдём лучшее из решений для всех таких пар.

Наибольшая общая подпоследовательность

Сводим к следующим подзадачам:

- Даны префиксы (начала) последовательностей a и b: $a_{1..i}$ и $b_{1..j}$.
- Найдём их наибольшую общую подпоследовательность.

Решение подзадачи для $a_{1..i}$ и $b_{1..j}$:

- Пусть оптимальное решение непусто (иначе всё очевидно).
- Рассмотрим последнее число в нём.
- Это число равно какому-то a_x $(1 \leqslant x \leqslant i)$.
- Это число также равно какому-то b_y $(1 \leqslant y \leqslant j)$.
- Значит, для получения оптимального решения нужно решить подзадачу для префиксов $a_{1..x-1}$ и $b_{1..y-1}$ и приписать к полученному решению это число (оно равно a_x и b_y).
- Если пар (x, y) несколько, можно рассматривать только максимальные x и y.

Наибольшая общая подпоследовательность

	b_j		**5**	**1**	**4**	**2**	**6**	**3**	**2**
a_i	$i \setminus j$	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0
2	2	0	0	0	0	2	0	0	2
4	3	0	0	0	2	0	0	0	0
6	4	0	0	0	0	0	3	0	0
3	5	0	0	0	0	0	0	4	0

Пример:

- $a = (1, 2, 4, 6, 3)$, $m = 5$
- $b = (5, 1, 4, 2, 6, 3, 2)$, $n = 7$

Наибольшая общая подпоследовательность

	b_j		**5**	**1**	**4**	**2**	**6**	**3**	**2**
a_i	$i \setminus j$	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0
2	2	0	0	0	0	2	0	0	2
4	3	0	0	0	2	0	0	0	0
6	4	0	0	0	0	0	3	0	0
3	5	0	0	0	0	0	0	4	0

- $f(i, j)$ — ответ для $a_{1..i}$ и $b_{1..j}$ при условии, что $a_i = b_j$
- База: $f(i, 0) = f(0, j) = 0$
- Ответ: $\max\limits_{i=0..m} \max\limits_{j=0..n} f(i, j)$
- Асимптотика: $O(m^2 n^2)$ времени, $O(mn)$ памяти

Наибольшая общая подпоследовательность

	b_j		**5**	**1**	**4**	**2**	**6**	**3**	**2**
a_i	$i \setminus j$	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0
2	2	0	0	0	0	2	0	0	2
4	3	0	0	0	2	0	0	0	0
6	4	0	0	0	0	0	3	0	0
3	5	0	0	0	0	0	0	4	0

```
for i := 1 upto m:
 for j := 1 upto n:
   if a[i] = b[j]:
     for u := 1 upto i - 1:
       for v := 1 upto j - 1:
         f[i][j] := max (f[i][j], f[u][v] + 1)
```

Наибольшая общая подпоследовательность

	b_j		**5**	**1**	**4**	**2**	**6**	**3**	**2**
a_i	$i \setminus j$	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
1	1	0	0	1	1	1	1	1	1
2	2	0	0	1	1	2	2	2	2
4	3	0	0	1	2	2	2	2	2
6	4	0	0	1	2	2	3	3	3
3	5	0	0	1	2	2	3	4	4

Пример:

- $a = (1, 2, 4, 6, 3)$, $m = 5$
- $b = (5, 1, 4, 2, 6, 3, 2)$, $n = 7$

Наибольшая общая подпоследовательность

	b_j		**5**	**1**	**4**	**2**	**6**	**3**	**2**
a_i	$i \setminus j$	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
1	1	0	0	1	1	1	1	1	1
2	2	0	0	1	1	2	2	2	2
4	3	0	0	1	2	2	2	2	2
6	4	0	0	1	2	2	3	3	3
3	5	0	0	1	2	2	3	4	4

- $g(i,j)$ — лучший из ответов в прямоугольнике $[0..i] \times [0..j]$
- База: $g(i,0) = g(0,j) = 0$
- Ответ: $g(m,n)$
- Асимптотика: $O(m^2n^2)$ времени (???), $O(mn)$ памяти

Наибольшая общая подпоследовательность

	b_j		**5**	**1**	**4**	**2**	**6**	**3**	**2**
a_i	$i \setminus j$	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
1	1	0	0	1	1	1	1	1	1
2	2	0	0	1	1	2	2	2	2
4	3	0	0	1	2	2	2	2	2
6	4	0	0	1	2	2	3	3	3
3	5	0	0	1	2	2	3	4	4

- $g(i, j)$ — лучший из ответов в прямоугольнике $[0..i] \times [0..j]$
- Все решения, в которых последнее число не является одновременно a_i и b_j, уже учтены либо в прямоугольнике $[0..i] \times [0..j-1]$, либо в прямоугольнике $[0..i-1] \times [0..j]$
- Асимптотика: $O(mn)$ времени, $O(mn)$ памяти

Наибольшая общая подпоследовательность

	b_j		**5**	**1**	**4**	**2**	**6**	**3**	**2**
a_i	$i \setminus j$	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
1	1	0	0	1	1	1	1	1	1
2	2	0	0	1	1	2	2	2	2
4	3	0	0	1	2	2	2	2	2
6	4	0	0	1	2	2	3	3	3
3	5	0	0	1	2	2	3	4	4

```
for i := 1 upto m:
 for j := 1 upto n:
   g[i][j] := max (g[i][j - 1], g[i - 1][j])
   if a[i] = b[j]:
     g[i][j] := max (g[i][j], g[i - 1][j - 1] + 1)
```

Оглавление

Редакционное расстояние

Определение:

- Редакционное расстояние (или расстояние Левенштейна) $d(s,t)$ между двумя строками s и t — это минимальное количество добавлений, удалений и замен символов, которые нужно сделать с s, чтобы получить t.

Пример:

- $d(\text{gets}, \text{goat}) = 3$: $\text{gets} \xrightarrow{-\mathtt{s}} \text{get} \xrightarrow{\mathtt{e}\to\mathtt{o}} \text{got} \xrightarrow{+\mathtt{a}} \text{goat}$.
  Другой способ: $\text{gets} \xrightarrow{\mathtt{e}\to\mathtt{o}} \text{gots} \xrightarrow{\mathtt{t}\to\mathtt{a}} \text{goas} \xrightarrow{\mathtt{s}\to\mathtt{t}} \text{goat}$.

Постановка задачи:

- По двум данным строкам требуется найти редакционное расстояние между ними.

Замечание:

- Строки s и t равноправны: к каждому действию есть обратное. Если выполнить обратные действия в обратном порядке, из t получится s.

Редакционное расстояние

Сводим к следующим подзадачам:

- Даны префиксы (начала) строк s и t: $s_{1..i}$ (префикс строки s длины i) и $t_{1..j}$ (префикс строки t длины j).
- Найдём редакционное расстояние между ними.

Решение подзадачи для $s_{1..i}$ и $t_{1..j}$:

- Если $s_i = t_j$, перейдём к решению подзадачи для $s_{1..i-1}$ и $t_{1..j-1}$.
- В противном случае оптимальная последовательность действий непуста, и либо с s_i, либо с t_j случилось хотя бы одно действие.
- Рассмотрим какое-нибудь из этих действий.
- Если это удаление, то это удаление символа s_i. Делаем его и переходим к подзадаче для $s_{1..i-1}$ и $t_{1..j}$.
- Если это добавление, то это добавление символа t_j. Делаем его и переходим к подзадаче для $s_{1..i}$ и $t_{1..j-1}$.
- Если это замена, то это замена символа s_i на символ t_j. Делаем её и переходим к подзадаче для $s_{1..i-1}$ и $t_{1..j-1}$.

Редакционное расстояние

Сводим к следующим подзадачам:

- Даны префиксы (начала) строк s и t: $s_{1..i}$ (префикс строки s длины i) и $t_{1..j}$ (префикс строки t длины j).
- Найдём редакционное расстояние между ними.

Решение подзадачи для $s_{1..i}$ и $t_{1..j}$:

- Если $s_i = t_j$, перейдём к решению подзадачи для $s_{1..i-1}$ и $t_{1..j-1}$.
- В противном случае оптимальная последовательность действий непуста, и либо с s_i, либо с t_j случилось хотя бы одно действие.
- Рассмотрим какое-нибудь из этих действий.
- Если это удаление, то это удаление символа s_i. Делаем его и переходим к подзадаче для $s_{1..i-1}$ и $t_{1..j}$.
- Если это добавление, то это добавление символа t_j. Делаем его и переходим к подзадаче для $s_{1..i}$ и $t_{1..j-1}$.
- Если это замена, то это замена символа s_i на символ t_j. Делаем её и переходим к подзадаче для $s_{1..i-1}$ и $t_{1..j-1}$.

Редакционное расстояние

Сводим к следующим подзадачам:

- Даны префиксы (начала) строк s и t: $s_{1..i}$ (префикс строки s длины i) и $t_{1..j}$ (префикс строки t длины j).
- Найдём редакционное расстояние между ними.

Решение подзадачи для $s_{1..i}$ и $t_{1..j}$:

- Если $s_i = t_j$, перейдём к решению подзадачи для $s_{1..i-1}$ и $t_{1..j-1}$.
- В противном случае оптимальная последовательность действий непуста, и либо с s_i, либо с t_j случилось хотя бы одно действие.
- Рассмотрим какое-нибудь из этих действий.
- Если это удаление, то это удаление символа s_i. Делаем его и переходим к подзадаче для $s_{1..i-1}$ и $t_{1..j}$.
- Если это добавление, то это добавление символа t_j. Делаем его и переходим к подзадаче для $s_{1..i}$ и $t_{1..j-1}$.
- Если это замена, то это замена символа s_i на символ t_j. Делаем её и переходим к подзадаче для $s_{1..i-1}$ и $t_{1..j-1}$.

Редакционное расстояние

Пример:

- $s =$ gets
- $t =$ goat

	t_j		g	o	a	t
s_i	$i \setminus j$	0	1	2	3	4
	0	0	1	2	3	4
g	1	1	0	1	2	3
e	2	2	1	1	2	3
t	3	3	2	2	2	2
s	4	4	3	3	3	3

- База: $d(i, 0) = i$, $d(0, j) = j$
- Ответ: $d(m, n)$, где m — длина строки s, а n — длина строки t
- Асимптотика: $O(mn)$ времени, $O(mn)$ памяти

Редакционное расстояние

Пример:

- $s = \texttt{gets}$
- $t = \texttt{goat}$

	t_j		g	o	a	t
s_i	$i \setminus j$	0	1	2	3	4
	0	0	1	2	3	4
g	1	1	0	1	2	3
e	2	2	1	1	2	3
t	3	3	2	2	2	2
s	4	4	3	3	3	3

```
for i := 1 upto m:
 for j := 1 upto n:
   if s[i] = t[j]:
     d[i][j] := d[i - 1][j - 1]
   else:
     d[i][j] := 1 + min (d[i - 1][j ],
                         d[i - 1][j - 1],
                         d[i ][j - 1])
```

Оглавление

Строка и шаблон

Определения:

- Шаблон — это строка, которая может содержать специальные символы '?' и '∗'.
- Обычному символу шаблона можно поставить в соответствие ровно один такой же символ строки.
- Специальному символу '?' можно поставить в соответствие ровно один любой символ строки.
- Специальному символу '∗' можно поставить в соответствие любую (в том числе и пустую) последовательность любых символов строки.
- Строка s длины m соответствует шаблону p длины n, если она разбивается на n частей так, что i-я часть строки соответствует i-му символу шаблона.

Строка и шаблон

Постановка задачи:

- По данным s и p нужно определить, соответствует ли строка s шаблону p.

Пример:

- $s = \texttt{button}$, $p = \texttt{?u*t*n}$
- Соответствие:

b	u	t	t	o	n
?	u	*	t	*	n

- Другой способ:

b	u		t	to	n
?	u	*	t	*	n

Строка и шаблон

Сводим к следующим подзадачам:

- Даны префиксы (начала) строки s и шаблона t: $s_{1..i}$ (префикс строки s длины i) и $p_{1..j}$ (префикс шаблона p длины j).
- Выясним, есть ли между ними соответствие.

Решение подзадачи для $s_{1..i}$ и $p_{1..j}$:

- Если p_j — обычный символ, сравним его с s_i и в случае равенства перейдём к решению подзадачи для $s_{1..i-1}$ и $p_{1..j-1}$.
- Если $p_j = ?$, перейдём к решению подзадачи для $s_{1..i-1}$ и $p_{1..j-1}$.
- Если $p_j = *$, соответствие есть тогда и только тогда, когда соответствие есть хотя бы в одной из подзадач для $s_{1..k}$ и $p_{1..j-1}$ при $0 \leqslant k \leqslant i$.

Замечание:

- Чтобы база была простой, допишем к обеим строкам спереди одинаковый специальный символ, например, #.

Строка и шаблон

Сводим к следующим подзадачам:

- Даны префиксы (начала) строки s и шаблона t: $s_{1..i}$ (префикс строки s длины i) и $p_{1..j}$ (префикс шаблона p длины j).
- Выясним, есть ли между ними соответствие.

Решение подзадачи для $s_{1..i}$ и $p_{1..j}$:

- Если p_j — обычный символ, сравним его с s_i и в случае равенства перейдём к решению подзадачи для $s_{1..i-1}$ и $p_{1..j-1}$.
- Если $p_j = ?$, перейдём к решению подзадачи для $s_{1..i-1}$ и $p_{1..j-1}$.
- Если $p_j = *$, соответствие есть тогда и только тогда, когда соответствие есть хотя бы в одной из подзадач для $s_{1..k}$ и $p_{1..j-1}$ при $0 \leqslant k \leqslant i$.

Замечание:

- Чтобы база была простой, допишем к обеим строкам спереди одинаковый специальный символ, например, #.

Строка и шаблон

	s_i	#	b	u	t	t	o	n
p_j	$j \setminus i$	1	2	3	4	5	6	7
#	1	1	0	0	0	0	0	0
?	2	0	1	0	0	0	0	0
u	3	0	0	1	0	0	0	0
*	4	0	0	1	1	1	1	1
t	5	0	0	0	1	1	0	0
*	6	0	0	0	1	1	1	1
n	7	0	0	0	0	0	0	1

- База: $f(0,0) = 1$, $f(i,0) = 0$ при $i > 0$, $f(0,j) = 0$ при $j > 0$
- Ответ: $f(m,n)$
- Асимптотика: $O(m^2 n)$ времени (???), $O(mn)$ памяти

Строка и шаблон

```
for i := 1 upto m:
 for j := 1 upto n:
   if p[i] = '*':
     f[i][j] := 0
     for k := 0 upto i:
       f[i][j] := f[i][j] or f[k][j - 1]
   else if p[i] = '?':
     f[i][j] := f[i - 1][j - 1]
   else:
     f[i][j] := f[i - 1][j - 1] and (s[i] = p[j])
```

- База: $f(0,0) = 1$, $f(i,0) = 0$ при $i > 0$, $f(0,j) = 0$ при $j > 0$
- Ответ: $f(m,n)$
- Асимптотика: $O(m^2 n)$ времени (???), $O(mn)$ памяти

Строка и шаблон

	s_i	#	b	u	t	t	o	n
p_j	$j \setminus i$	1	2	3	4	5	6	7
#	1	1	0	0	0	0	0	0
?	2	0	1	0	0	0	0	0
u	3	0	0	1	0	0	0	0
*	4	0	0	1	1	1	1	1
t	5	0	0	0	1	1	0	0
*	6	0	0	0	1	1	1	1
n	7	0	0	0	0	0	0	1

При $p_j = *$:

- $f(i,j) = f(0,j-1)$ or $\ldots$ or $f(i-1,j-1)$ or $f(i,j-1)$
- $f(i-1,j) = f(0,j-1)$ or $\ldots$ or $f(i-1,j-1)$
- Значит, $f(i,j) = f(i-1,j)$ or $f(i,j-1)$

Строка и шаблон

```
for i := 1 upto m:
 for j := 1 upto n:
   if p[i] = '*':
     f[i][j] := f[i - 1][j] or f[i][j - 1]
   else if p[i] = '?':
     f[i][j] := f[i - 1][j - 1]
   else:
     f[i][j] := f[i - 1][j - 1] and (s[i] = p[j])
```

- База: $f(0,0) = 1$, $f(i,0) = 0$ при $i > 0$, $f(0,j) = 0$ при $j > 0$
- Ответ: $f(m,n)$
- Асимптотика: $O(mn)$ времени, $O(mn)$ памяти

Всё.