

# с/к “Эффективные алгоритмы”

## Лекция 2: Приближенные алгоритмы

А. Куликов

Computer Science клуб при ПОМИ  
<http://logic.pdmi.ras.ru/~infclub/>



# План лекции

- 1 Кратчайшая надпоследовательность
  - Пример работы алгоритма

# Приближенный алгоритм

## Определение

# Приближенный алгоритм

## Определение

- Приближенный алгоритм  $A$  имеет **относительную оценку точности** (relative performance guarantee)  $\alpha$ , если для любого входа  $I$  выполняется неравенство

# Приближенный алгоритм

## Определение

- Приближенный алгоритм  $A$  имеет **относительную оценку точности** (relative performance guarantee)  $\alpha$ , если для любого входа  $I$  выполняется неравенство
  - ▶  $A(I)/OPT(I) \geq \alpha$  для максимизационной задачи;

# Приближенный алгоритм

## Определение

- Приближенный алгоритм  $A$  имеет **относительную оценку точности** (relative performance guarantee)  $\alpha$ , если для любого входа  $I$  выполняется неравенство
  - ▶  $A(I)/OPT(I) \geq \alpha$  для максимизационной задачи;
  - ▶  $A(I)/OPT(I) \leq \alpha$  для минимизационной задачи.

# Приближенный алгоритм

## Определение

- Приближенный алгоритм  $A$  имеет **относительную оценку точности** (relative performance guarantee)  $\alpha$ , если для любого входа  $I$  выполняется неравенство
  - ▶  $A(I)/OPT(I) \geq \alpha$  для максимизационной задачи;
  - ▶  $A(I)/OPT(I) \leq \alpha$  для минимизационной задачи.
- Такие алгоритмы мы называем  $\alpha$ -приближенными.

# Приближенный алгоритм

## Определение

- Приближенный алгоритм  $A$  имеет **относительную оценку точности** (relative performance guarantee)  $\alpha$ , если для любого входа  $I$  выполняется неравенство
  - ▶  $A(I)/OPT(I) \geq \alpha$  для максимизационной задачи;
  - ▶  $A(I)/OPT(I) \leq \alpha$  для минимизационной задачи.
- Такие алгоритмы мы называем  $\alpha$ -приближенными.

## Замечание

Для оптимизационных задач  $\alpha \leq 1$ , для минимизационных —  $\alpha \geq 1$ .  
Алгоритм тем лучше, чем ближе  $\alpha$  к 1.

# План лекции

- 1 Кратчайшая надпоследовательность
  - Пример работы алгоритма

# Задача о кратчайшей общей надпоследовательности

## Определение

# Задача о кратчайшей общей надпоследовательности

## Определение

- Дано множество строк  $\{s_1, \dots, s_n\}$ .

# Задача о кратчайшей общей надпоследовательности

## Определение

- Дано множество строк  $\{s_1, \dots, s_n\}$ .
- **Задача о кратчайшей общей надпоследовательности** (shortest common superstring) заключается в нахождении самой короткой строки  $u$ , которая содержит в качестве подстроки каждую из  $s_j$ .

# Задача о кратчайшей общей надпоследовательности

## Определение

- Дано множество строк  $\{s_1, \dots, s_n\}$ .
- **Задача о кратчайшей общей надпоследовательности** (shortest common superstring) заключается в нахождении самой короткой строки  $u$ , которая содержит в качестве подстроки каждую из  $s_j$ .

## Факт

Задача является NP-трудной.

# Задача о покрытии множествами

## Определение

# Задача о покрытии множествами

## Определение

- Дано множество  $U = \{u_1, \dots, u_n\}$  и семейство его подмножеств  $\mathcal{F} = \{S_1, \dots, S_k\}$ ,  $S_i \subseteq U$ .

# Задача о покрытии множествами

## Определение

- Дано множество  $U = \{u_1, \dots, u_n\}$  и семейство его подмножеств  $\mathcal{F} = \{S_1, \dots, S_k\}$ ,  $S_i \subseteq U$ .
- В сумме все подмножества покрывают  $U$ :  $U = \bigcup_{S \in \mathcal{F}} S$ .

# Задача о покрытии множествами

## Определение

- Дано множество  $U = \{u_1, \dots, u_n\}$  и семейство его подмножеств  $\mathcal{F} = \{S_1, \dots, S_k\}$ ,  $S_i \subseteq U$ .
- В сумме все подмножества покрывают  $U$ :  $U = \bigcup_{S \in \mathcal{F}} S$ .
- Каждому подмножеству  $S_i$  сопоставлена некоторая неотрицательная стоимость  $p_i \geq 0$ .

# Задача о покрытии множествами

## Определение

- Дано множество  $U = \{u_1, \dots, u_n\}$  и семейство его подмножеств  $\mathcal{F} = \{S_1, \dots, S_k\}$ ,  $S_i \subseteq U$ .
- В сумме все подмножества покрывают  $U$ :  $U = \bigcup_{S \in \mathcal{F}} S$ .
- Каждому подмножеству  $S_i$  сопоставлена некоторая неотрицательная стоимость  $p_i \geq 0$ .
- **Задача о покрытии множествами** (set cover problem) заключается в нахождении набора подмножеств, покрывающего все множество  $U$  и имеющего минимальный возможный вес.

# Задача о покрытии множествами

## Определение

- Дано множество  $U = \{u_1, \dots, u_n\}$  и семейство его подмножеств  $\mathcal{F} = \{S_1, \dots, S_k\}$ ,  $S_i \subseteq U$ .
- В сумме все подмножества покрывают  $U$ :  $U = \bigcup_{S \in \mathcal{F}} S$ .
- Каждому подмножеству  $S_i$  сопоставлена некоторая неотрицательная стоимость  $p_i \geq 0$ .
- **Задача о покрытии множествами** (set cover problem) заключается в нахождении набора подмножеств, покрывающего все множество  $U$  и имеющего минимальный возможный вес.

## Теорема

Существует  $H_n$ -приближенный алгоритм для задачи о покрытии множествами.

# Сведение к задаче о покрытии множествами

## Сведение к задаче о покрытии множествами

- НУО, среди строк нет подстрок друг друга

## Сведение к задаче о покрытии множествами

- НУО, среди строк нет подстрок друг друга
- для всех  $i, j, k$ , для которых суффикс строки  $s_i$  длины  $k$  совпадает с префиксом строки  $s_j$  длины  $k$ , построим строку  $w_{ijk} = s_i \cdot s_j[k + 1..|s_j|]$ :

## Сведение к задаче о покрытии множествами

- НУО, среди строк нет подстрок друг друга
- для всех  $i, j, k$ , для которых суффикс строки  $s_i$  длины  $k$  совпадает с префиксом строки  $s_j$  длины  $k$ , построим строку  $w_{ijk} = s_i \cdot s_j[k + 1..|s_j|]$ :

$$s_6 = \underbrace{abcab}_X \underbrace{bbca}_Y, s_2 = \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

## Сведение к задаче о покрытии множествами

- НУО, среди строк нет подстрок друг друга
- для всех  $i, j, k$ , для которых суффикс строки  $s_i$  длины  $k$  совпадает с префиксом строки  $s_j$  длины  $k$ , построим строку  $w_{ijk} = s_i \cdot s_j[k + 1..|s_j|]$ :

$$s_6 = \underbrace{abcab}_X \underbrace{bbca}_Y, s_2 = \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

$$w_{6,2,4} = \underbrace{abcab}_X \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

## Сведение к задаче о покрытии множествами

- НУО, среди строк нет подстрок друг друга
- для всех  $i, j, k$ , для которых суффикс строки  $s_i$  длины  $k$  совпадает с префиксом строки  $s_j$  длины  $k$ , построим строку  $w_{ijk} = s_i \cdot s_j[k + 1..|s_j|]$ :

$$s_6 = \underbrace{abcab}_X \underbrace{bbca}_Y, s_2 = \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

$$w_{6,2,4} = \underbrace{abcab}_X \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

- для любой строки  $s$  определим  $\text{set}(s) = \{s_i | s_i \text{ — подстрока } s\}$

## Сведение к задаче о покрытии множествами

- НУО, среди строк нет подстрок друг друга
- для всех  $i, j, k$ , для которых суффикс строки  $s_i$  длины  $k$  совпадает с префиксом строки  $s_j$  длины  $k$ , построим строку  $w_{ijk} = s_i \cdot s_j[k + 1..|s_j|]$ :

$$s_6 = \underbrace{abcab}_X \underbrace{bbca}_Y, s_2 = \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

$$w_{6,2,4} = \underbrace{abcab}_X \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

- для любой строки  $s$  определим  $\text{set}(s) = \{s_i | s_i \text{ — подстрока } s\}$
- стоимость множества  $\text{set}(s)$  положим равной длине строки  $s$

## Сведение к задаче о покрытии множествами

- НУО, среди строк нет подстрок друг друга
- для всех  $i, j, k$ , для которых суффикс строки  $s_i$  длины  $k$  совпадает с префиксом строки  $s_j$  длины  $k$ , построим строку  $w_{ijk} = s_i \cdot s_j[k + 1..|s_j|]$ :

$$s_6 = \underbrace{abcab}_X \underbrace{bbca}_Y, s_2 = \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

$$w_{6,2,4} = \underbrace{abcab}_X \underbrace{bbca}_Y \underbrace{aacbcac}_Z$$

- для любой строки  $s$  определим  $\text{set}(s) = \{s_i | s_i \text{ — подстрока } s\}$
- стоимость множества  $\text{set}(s)$  положим равной длине строки  $s$
- вход задачи о покрытии множествами:  $U = \{s_1, \dots, s_n\}$ ,  
 $\mathcal{F} = \{\text{set}(s_i)\} \cup \{\text{set}(w_{ijk})\}$

# Сведение к задаче о покрытии множествами (продолжение)

## Сведение к задаче о покрытии множествами (продолжение)

- итак, алгоритм возвращает нам некоторые множества

## Сведение к задаче о покрытии множествами (продолжение)

- итак, алгоритм возвращает нам некоторые множества
- мы сливаем соответствующие им строки и получаем строку, содержащую все  $S_j$

## Сведение к задаче о покрытии множествами (продолжение)

- итак, алгоритм возвращает нам некоторые множества
- мы сливаем соответствующие им строки и получаем строку, содержащую все  $S_j$
- осталось показать, что построенный нами алгоритм является  $2H_n$ -приближенным

## Сведение к задаче о покрытии множествами (продолжение)

- итак, алгоритм возвращает нам некоторые множества
- мы сливаем соответствующие им строки и получаем строку, содержащую все  $S_j$
- осталось показать, что построенный нами алгоритм является  $2H_n$ -приближенным
- для этого достаточно доказать приведенную ниже лемму

## Сведение к задаче о покрытии множествами (продолжение)

- итак, алгоритм возвращает нам некоторые множества
- мы сливаем соответствующие им строки и получаем строку, содержащую все  $s_j$
- осталось показать, что построенный нами алгоритм является  $2H_n$ -приближенным
- для этого достаточно доказать приведенную ниже лемму

### Лемма

Стоимость оптимального решения полученной задачи о покрытии множествами не более чем в два раза хуже длины решения исходной задачи о кратчайшей надпоследовательности.

# Доказательство леммы

## Доказательство

## Доказательство леммы

### Доказательство

- перенумеруем строки  $s_j$  в порядке их вхождения в (какую-нибудь) оптимальную строку  $s$

# Доказательство леммы

## Доказательство

- перенумеруем строки  $s_j$  в порядке их вхождения в (какую-нибудь) оптимальную строку  $s$
- ясно, что вхождение каждой строки начинается и заканчивается строго позже предыдущей

# Доказательство леммы

## Доказательство

- перенумеруем строки  $s_j$  в порядке их вхождения в (какую-нибудь) оптимальную строку  $s$
- ясно, что вхождение каждой строки начинается и заканчивается строго позже предыдущей
- разделим строки на блоки следующим образом: в первый блок берем  $s_1$  и все  $s_j$ , первые вхождения которых начинаются до конца первого вхождения  $s_1$

# Доказательство леммы

## Доказательство

- перенумеруем строки  $s_j$  в порядке их вхождения в (какую-нибудь) оптимальную строку  $s$
- ясно, что вхождение каждой строки начинается и заканчивается строго позже предыдущей
- разделим строки на блоки следующим образом: в первый блок берем  $s_1$  и все  $s_j$ , первые вхождения которых начинаются до конца первого вхождения  $s_1$
- сделаем из первой и последней строки этого блока общую строку — получим строку  $w_{1,j,k}$

# Доказательство леммы

## Доказательство

- перенумеруем строки  $s_j$  в порядке их вхождения в (какую-нибудь) оптимальную строку  $s$
- ясно, что вхождение каждой строки начинается и заканчивается строго позже предыдущей
- разделим строки на блоки следующим образом: в первый блок берем  $s_1$  и все  $s_j$ , первые вхождения которых начинаются до конца первого вхождения  $s_1$
- сделаем из первой и последней строки этого блока общую строку — получим строку  $w_{1,j,k}$
- следующий блок строим, начиная с первой строки, не попавшей в первый блок, и т.д.

# Доказательство леммы

## Доказательство

- перенумеруем строки  $s_j$  в порядке их вхождения в (какую-нибудь) оптимальную строку  $s$
- ясно, что вхождение каждой строки начинается и заканчивается строго позже предыдущей
- разделим строки на блоки следующим образом: в первый блок берем  $s_1$  и все  $s_j$ , первые вхождения которых начинаются до конца первого вхождения  $s_1$
- сделаем из первой и последней строки этого блока общую строку — получим строку  $w_{1,j,k}$
- следующий блок строим, начиная с первой строки, не попавшей в первый блок, и т.д.
- в итоге, получаем некоторое решение задачи о покрытии множествами

## Доказательство леммы (продолжение)

Доказательство

## Доказательство леммы (продолжение)

### Доказательство

- достаточно показать, что стоимость данного конкретного решения не превосходит удвоенной длины  $s$

## Доказательство леммы (продолжение)

### Доказательство

- достаточно показать, что стоимость данного конкретного решения не превосходит удвоенной длины  $s$
- это верно, поскольку каждый символ строки  $s$  входит не более чем в два блока □

## Доказательство леммы (продолжение)

### Доказательство

- достаточно показать, что стоимость данного конкретного решения не превосходит удвоенной длины  $s$
- это верно, поскольку каждый символ строки  $s$  входит не более чем в два блока □

### Упражнение

Докажите последнее утверждение.

# План лекции

- 1 Кратчайшая надпоследовательность
  - Пример работы алгоритма

# Пример работы алгоритма

## Пример работы алгоритма

- допустим, на вход даны следующие строки:  
 $s_1 = abc$ ,  $s_2 = bac$ ,  $s_3 = bcd$ ,  $s_4 = cde$

## Пример работы алгоритма

- допустим, на вход даны следующие строки:

$$s_1 = abc, s_2 = bac, s_3 = bcd, s_4 = cde$$

- вычисляем  $w_{ijk}$ :

$$w_{132} = abcd, w_{141} = abcde, w_{241} = bacde, w_{342} = bcde$$

## Пример работы алгоритма

- допустим, на вход даны следующие строки:

$$s_1 = abc, s_2 = bac, s_3 = bcd, s_4 = cde$$

- вычисляем  $w_{ijk}$ :

$$w_{132} = abcd, w_{141} = abcde, w_{241} = bacde, w_{342} = bcde$$

- вычисляем  $\text{set}(s_i)$  и  $\text{set}(w_{ijk})$ , а также стоимости:

$$S_1 = \text{set}(s_1) = \{s_1\}, p_1 = 3,$$

$$S_2 = \text{set}(s_2) = \{s_2\}, p_2 = 3,$$

$$S_3 = \text{set}(s_3) = \{s_3\}, p_3 = 3,$$

$$S_4 = \text{set}(s_4) = \{s_4\}, p_4 = 3,$$

$$S_5 = \text{set}(w_{132}) = \{s_1, s_3\}, p_5 = 4$$

$$S_6 = \text{set}(w_{141}) = \{s_1, s_3, s_4\}, p_6 = 5,$$

$$S_7 = \text{set}(w_{241}) = \{s_2, s_4\}, p_7 = 5,$$

$$S_8 = \text{set}(w_{342}) = \{s_3, s_4\}, p_8 = 4$$

# Пример работы алгоритма

## Пример работы алгоритма

- итак, входными данными задачи о покрытии множествами являются универсальное множество  $U = \{s_i\}_{i \in [1..4]}$ , множество его подмножеств  $\mathcal{F} = \{S_i\}_{i \in [1..8]}$ , где каждому подмножеству  $S_i$  сопоставлена стоимость  $p_i$

## Пример работы алгоритма

- итак, входными данными задачи о покрытии множествами являются универсальное множество  $U = \{s_i\}_{i \in [1..4]}$ , множество его подмножеств  $\mathcal{F} = \{S_i\}_{i \in [1..8]}$ , где каждому подмножеству  $S_i$  сопоставлена стоимость  $p_i$
- алгоритм последовательно добавляет в покрытие множества  $S_6$  и  $S_2$

## Пример работы алгоритма

- итак, входными данными задачи о покрытии множествами являются универсальное множество  $U = \{s_i\}_{i \in [1..4]}$ , множество его подмножеств  $\mathcal{F} = \{S_i\}_{i \in [1..8]}$ , где каждому подмножеству  $S_i$  сопоставлена стоимость  $p_i$
- алгоритм последовательно добавляет в покрытие множества  $S_6$  и  $S_2$
- поскольку  $S_6 = \text{set}(w_{141})$  и  $S_2 = \text{set}(s_2)$ , выходом нашего алгоритма будет конкатенация строк  $w_{141}$  и  $s_2$ , то есть строка  $abcdebac$ , которая в данном конкретном примере является оптимальной

Что мы узнали за сегодня?

Что мы узнали за сегодня?

# Что мы узнали за сегодня?

## Что мы узнали за сегодня?

- $2H_n$ -приближенный алгоритм для задачи о кратчайшей надпоследовательности.

Спасибо за внимание!