Умножение матриц над полем GF(2) методом 'четырёх русских'

Калинин Михаил, 303 группа

25 октября 2022 г.

Содержание

1 Вступление **3**

2 Описание алгоритма **3**

3 Таблица времени работы для разных n. Версия 1 **3**

4 Таблица времени работы для разных n. Версия 2 **3**

5 Вычисление параметра α для версии 1 **3**

6 Вычисление параметра α для версии 2 **3**

7 Выводы **4**

1 Вступление

Существует множество алгоритмов, использующих перемножение двоичных матриц. Например:

1. Вычисление транзитивного замыкания графа
2. Расчёт расстояния редактирования
3. Выравнивание последовательности
4. Вычисление индекса для двоичного сопоставления с шаблоном

Чтобы ускорить их выполнение, можно использовать метод 'Четырёх русских', разработанный В. Л. Арлазаровым, Е. А. Диницем, М. А. Кронродом и И. А. Фараджевым в 1970 году. Он позволяет выполнить алгоритм в $log_2 n$ раз быстрее.

2 Описание алгоритма

Даны две матрицы $A \in P^{m \times n}$ и $B \in P^{n \times k}$, где P - поле по модулю 2. Выберем число $k = \lfloor log_2(n) \rfloor$. Для всех возможных пар двоичных векторов длины k подсчитаем и запомним их скалярное произведение по модулю 2. Первую матрицу разделим каждую её строку на куски размера k. Если строка не делится нацело, до добавим столбы нулей, чтобы делилась. Получим матрицу $A'_{n \times \lceil \frac{n}{k} \rceil}$. Аналогично сделаем со столбцами матрицы B. Получим матрицу $B'_{\lceil \frac{n}{k} \rceil \times n}$. Теперь посчитаем произведение новых матриц A' и B', воспользовавшись посчитанными скалярными произведениями.

3 Таблица времени работы для разных n. Версия 1

Размеры матриц	1024	2048	4096	8192	16384
Время работы, сек	3	25	251	2499	>21600

К сожалению, unsigned int перполнился и замерить время для случая 16384 не удалось. По ощущениям скажу, что точно больше 21600.

4 Таблица времени работы для разных n. Версия 2

Размеры матриц	1024	2048	4096	8192	16384
Время работы, сек	2	20	182	1395	11363

5 Вычисление параметра α для версии 1

При увличении с 1024 до 2048 получаем увеличение времени работы примерно в 8 раз. При увеличении с 2048 до 4096 - примерно 10 раз. При увеличении с 8192 до 16384 примерно в 9-10 раз. При увеличении с 8192 до 16384 время работы увеличивается в 8 раз. Получаем в среднем увеличение работы в 9 раз, при увеличении размеров матрицы в 2 раза. Значит, время работы увеличивается примерно в $log_2 9 = 3,17$ раз. А значит, алгоритм имеет сложность $O(n^{3.17})$.

6 Вычисление параметра α для версии 2

$\alpha = log_2(8.725) = 3.13$

7 Выводы

Как видим, ассимптотика хоть и не сильно улучшилась, зато мы всё равно получили сильный выйгрышь по скорости вычислений после добавлениям блочности умножения, использования коэффициента 6. Можно поиграть с оптимизациями и попробовать поставить другой коэффициент. Кроме того вижу простор для улучшения в работе с битовыми данными: использование BitPtr, в силу его реализации, не кажется мне эффективным.