

ПАРАЛЛЕЛЬНОЕ РЕШЕНИЕ ЛИНЕЙНЫХ СИСТЕМ НАД КОНЕЧНЫМИ ПОЛЯМИ И ОПТИМАЛЬНОЕ УМНОЖЕНИЕ МАТРИЧНЫХ ПОЛИНОМОВ

И.В. Оселедец

Большие линейные системы (порядка десятков или даже сотен миллионов) над конечными полями возникают во многих задачах криптографии. Например, при взломе криптосистемы RSA требуется разлагать очень большое натуральное число на множители. Наиболее эффективным алгоритмом является алгоритм решета числового поля (Number field sieve) одним из важнейших шагов которого является решение большой разреженной линейной системы над полем $GF(2)$. Для рекордного на данный момент вычисления размер матрицы составлял примерно 6×10^7 . Линейные системы над конечными полями с большим количеством элементов возникают в задаче дискретного логарифмирования. Для разложения действительно больших чисел на множители необходимо использовать огромные вычислительные ресурсы - терафлопные кластера, и все технологии GRID-вычислений (одними из первых задач, решённых с помощью распределённых вычислений, были задачи факторизации).

Мы сосредоточим своё внимание на линейных системах над полем $GF(2)$. Оказывается, что для них возникает очень много новых теоретических и практических проблем. В отличие от вещественного случая, при решении итерационным методом, число итераций будет порядка размера матрицы N , поэтому число операций становится порядка $O(N^2)$, что делает решение очень трудоёмким. Поэтому необходимо использовать параллельные вычисления. Существуют два метода решения линейных систем над полем $GF(2)$ - метод Ланцоша-Монтгомери и метод Копперсмита.

Метод Копперсмита более сложный в реализации, однако существенно более параллельный. Он состоит из двух шагов. На первом шаге насчитываются произведения вида

$$a_i = X^T A^i Y, \quad i = 0, \dots, N/n + N/m + 1,$$

где $X \in F^{N \times n}$, $Y \in F^{N \times m}$, где n, m - некоторые небольшие числа (блочность), кратные длине машинного слова (обычно 64). Этот шаг полностью параллелен и может выполняться даже на географически отдалённых кластерах.

После этого на втором шаге необходимо найти вектор из ядра блочно-ганкелевой матрицы $H = [a_{i+j}]$. Быстрый алгоритм, предложенный Копперсмитом, требует $N^2/8$ операций. Для $N \sim 10^8$ время счёта такого алгоритма будет порядка года. Поэтому нужно реализовывать *супербыстрый* вариант метода Копперсмита, который требует $O(N \log N)$ операций. Базовым ингредиентом является быстрое умножение матричных полиномов вида $c(x) = a(x) b(x)$, где $a(x), b(x)$ - матричные полиномы соответствующей размерности над полем $GF(2)$. Возникает вопрос об оптимальных (или очень хороших) алгоритмах умножения таких полиномов. Самый известный пример - классический алгоритм Карацубы для умножения линейных полиномов за 3 умножения. Обобщение этого результата для полиномов большего порядка получено недавно в 2005 году Монтгомери. Для полиномов 2 степени требуется 6 умножений коэффициентов, для 4 - 13, 5 - 17. В нашей работе получены оптимальные алгоритмы для больших степеней. Этот вопрос можно свести к задаче трилинейной аппроксимации некоторого трёхмерного массива (тензора). Были построены способы получения таких представлений, и построены оптимальные алгоритмы для умножения полиномов вплоть до степени 1000. В частности, для $n=1000$ число умножений для нового алгоритма меньше в 10 раз.