УДК 519.688

В. В. Перевощиков

ОБ ОПТИМАЛЬНОЙ ПЕРЕСЫЛКЕ ОСОБЫХ ТОЧЕК В ПАРАЛЛЕЛЬНОМ РО-МЕТОДЕ ПОЛЛАРДА

Рассмотрены проблемы разработки и реализации эффективного параллельного алгоритма дискретного логарифмирования в группе точек эллиптической кривой, основанного на ро-методе Полларда, в модели вычислений SPMD с использованием технологии обмена сообщений. Исследовано, при каком количестве центральных процессоров и какой доле особых точек, на которых осуществляется этот поиск, ожидаемое время работы алгоритма будет оптимальным при ограничениях на доступную память, а также дан анализ результатов тестирования разработанного программного комплекса.

Problems of development and realization of an efficient parallelized algorithm to solve elliptic curve discrete logarithm problem (ECDLP) based on Pollard's rho-method in the computational model SPMD using technology of message passing are considered. It is researched how many central processes are needed and what proportion of distinguished points is needed to be set to optimize an expected working time of this algorithm under the constraints of available memory; analyses the testing results of the developed program.

Ключевые слова: дискретный логарифм, ро-метод Полларда, эллиптические кривые, параллельное программирование, случайное блуждание.

Key words: discrete logarithm, Pollard's rho-method, elliptic curves, parallel programming, random walk.

Введение

Сегодня не существует эффективных алгоритмов дискретного логарифмирования в группе точек произвольной эллиптической кривой над произвольным конечным полем. В этом случае наиболее эффективен недетерминированный ро-метод Полларда, пригодный для дискретного логарифмирования в произвольной конечной циклической группе, имеющий тем не менее экспоненциальную сложность. Естественным способом улучшения времени работы этого алгоритма является использование параллельных вычислений.

Модель обмена сообщений — наиболее широко используемой модель параллельного программирования. Системы обмена сообщений реализуют модель SPMD (Single Program Multiple Data). В рассматриваемой модели каждый процесс выполняет одну и ту же программу, но работает с разными данными и имеет доступ лишь к части памяти вычислительной системы.

Авторы [1] предлагают использовать m процессов для генерации случайных блужданий общей итерационной функцией, а поиск коллизии осуществлять лишь на некотором небольшом подмножестве особых точек группы, точки из которой следует отправлять на обработку и хранение центральному процессу. При этом если доля особых точек очень мала, то это может сильно замедлить поиск коллизии и даже увеличить вероятность неуспешного завершения работы алгоритма. Иначе при другом порядке группы потребуется большее количество памяти, чем имеется в распоряжении центрального процесса.

Нами рассмотрены следующие вопросы: когда эффективно использовать несколько центральных процессов для хранения и обработки данных; какая доля особых точек оптимизирует ожидаемое время работы алгоритма. Теоретические результаты сравнивались с экспериментальными, полученными при тестировании программного комплекса.

1. Постановка задачи и описание метода

Пусть $< G, +, \Theta >$ — конечная группа, записанная в аддитивной форме (группа точек эллиптической кривой над некоторым полем), $< P >$ — циклическая подгруппа точки $P \in G$ порядка n, $Q \in < P >$. Тогда существует такое целое число $0 \leqslant x < n$, что

$$xP = Q. \qquad (1)$$

Задача нахождения такого числа x называется *дискретным логарифмированием* в группе $< P >$. Если n — не простое число, то решение (1) сводится к аналогичной задаче в группе простого порядка с помощью алгоритма Полига — Хеллмана и китайской теоремы об остатках. Если же n — простое, то решать уравнение вида (1) можно с помощью ро-метода Полларда. Суть метода

состоит в следующем. Пусть $f: < P > \to < P >$ — случайная функция, такая, что $f(R) = aR + bQ$, где $0 \leqslant a, b < n$. Начиная с некоторой случайной точки R_0 генерируется последовательность

$$R_{i+1} = f(R_i), i = 0, 1, \ldots, \qquad (2)$$

члены которой принимают лишь конечное число значений. Поэтому по принципу Дирихле существуют такие индексы $0 \leqslant i < j \leqslant n$, что $R_i = R_j$. Это называется *коллизией*. При этом $R_{i+k} = R_{j+k}$ для любого k. Это означает, что последовательность ведет себя циклически, начиная с некоторого члена. Если найдена коллизия, то получено уравнение

$$a_i + b_i x \equiv a_j + b_j x \pmod{n}, i \neq j. \qquad (3)$$

В случае, когда n — простое число, сравнение (3) всегда имеет единственное решение, если $b_i \neq b_j$. Решение (3) является в этом случае решением (1). Доказано, что ожидаемое число итераций до появления коллизии в последовательности равно примерно $\sqrt{\pi n / 2}$. В работе [1] показано, что если генерировать m последовательностей (2) с итерационной функцией f, а коллизию искать в совокупности сгенерированных членов, то из $R_i^{(u)} = R_j^{(v)}$ при $i \neq j$ следует

$$R_{i+k}^{(u)} = R_{j+k}^{(v)}. \qquad (4)$$

Если $u = v$, то последовательность ведет себя циклическим образом, в противном случае две последовательности совпадают, начиная с некоторых индексов. Искать коллизию среди всех сгенерированных точек стало бы накладно с точки зрения хранения, обработки и пересылки данных. Поэтому в работе [1] предложено искать коллизию лишь на некотором множестве $C \subset < P >$. Пусть $\theta = \frac{|C|}{n}$ — доля особых точек, на которых следует искать коллизию. Если возникла коллизия, то в силу выражения (4) зацикливание последовательности или совпадение двух последовательностей можно обнаружить, когда будет сгенерирована особая точка. Величина расстояния между двумя особыми точками блуждающей последовательности геометрически распределена, и ожидаемое значение этой величины равно $1/\theta$. Если T — количество точек, которые сгенерировал процесс до обнаружения коллизии, то $E[T] = \frac{1}{m}\sqrt{\frac{\pi n}{2}} + \frac{1}{\theta}$.

Как показано в работе [2], ожидаемое время работы алгоритма можно увеличить в $\sqrt{2}$ раз, если использовать групповой автоморфизм $\psi: < P > \to < P >$, $\psi(R) = -R$. Этот автоморфизм имеет порядок 2. Можно считать, что точки R_1 и R_2 эквивалентны, если $\psi^j(R_1) = R_2$. Это отношение эквивалентности разбивает группу $< P >$ на $\lceil n/2 \rceil$ классов эквивалентности. В каждом классе можно выделить главного представителя. Например, в группе точек эллиптической кривой это может быть точка, координаты которой меньше в лексикографическом смысле. Тогда ожидаемое время работы алгоритма

$$E[T] = \frac{\sqrt{\pi n}}{2m} + \frac{1}{\theta}. \qquad (5)$$

2. О хранении и обработке данных

Оценка количества генерируемых особых точек за время работы алгоритма приведена в работе [3]. Поскольку величина расстояния между двумя особыми точками блуждающей последовательности геометрически распределена с параметром θ, то можно считать, что в среднем каждые $1/\theta$ итераций каждым процессом генерируется особая точка. Число точек, генерируемых до коллизии, и расстояния между двумя точками независимы, поэтому можно считать, что ожидаемое количество генерируемых всеми процессами точек до возникновения коллизии равно

$$E[S] = m\theta E[T] = (\theta\sqrt{\pi n})/2 + m. \qquad (6)$$

Для хранения этих точек и поиска коллизии рекомендуется использовать структуру сбалансированных деревьев, например АВЛ-деревьев. В работе [4] доказано, что высота такого дерева из r элементов не превосходит величины $\log_\varphi 2 \cdot \log_2(r+1) \approx 1,44 \log_2 r$, где φ — золотое сечение.

3. Об определении оптимальных параметров алгоритма

Выбор доли особых точек стоит производить из тех соображений, чтобы минимизировать ожидаемое время работы (5), но в то же время быть в готовности хранить количество данных, задаваемых формулой (6). Пусть p — общее количество процессов, причем x из них являются центральными, осуществляющими поиск коллизии. Также допустим, что α — среднее значение времени одной итерации ро-метода, β — среднее значение времени пересылки точки центральному процессору, M — количество точек, которые способен хранить каждый процесс, $A = \frac{\sqrt{\pi n}}{2}$. Тогда требуется решить следующую задачу оптимизации по минимизации времени работы:

$$E[T](x,\theta) = \alpha\left(\frac{A}{p-x} + \frac{1}{\theta}\right) + \beta\frac{A\theta}{p-x} \to \min,$$

$$A\theta \leqslant Mx,\ \ 0 < \theta < 1,\ 0 < x\ \ 0 < x \le p-1\ \ \ p-1.$$

Решением данной задачи являются

$$x_{\min} = \frac{p}{1+\sqrt{M + \frac{M^2 p}{A}\cdot\frac{\beta}{\alpha}}},\ \theta_{\min} = \frac{p}{A}\cdot\frac{1}{\frac{1}{M} + \sqrt{\frac{1}{M} + \frac{p}{A}\cdot\frac{\beta}{\alpha}}}.$$

Если $p < \sqrt{M}$, что соответствует практической ситуации, то $x_{\min} < 1$, то есть достаточно одного центрального процессора для хранения данных.

Пусть в распоряжении центрального процессора имеется 1 Гб памяти и на хранение данных, необходимых для поиска коллизии и решения уравнения (3), требуется не более 65 байт, тогда $M = 2^{24}$, $p = 100$, т.е. оптимальное число особых бит равно 10.

На рисунке 1 представлен график зависимости ожидаемого времени работы от числа особых бит s в группе порядка 43 бита. Минимум достигается при $s \approx 10$. На рисунке 2 изображен график зависимости ожидаемого времени работы от числа особых бит s в группе порядка 71 бит. Минимум ожидаемого времени работы в этом случае достигается при $s \approx 16$.

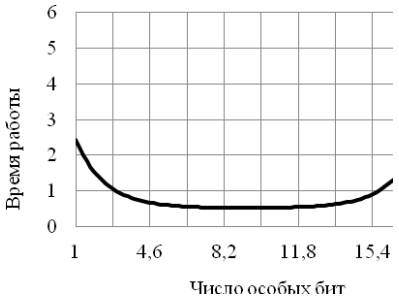

Рис. 1. Зависимость ожидаемого времени работы от числа особых бит в группе порядка 43

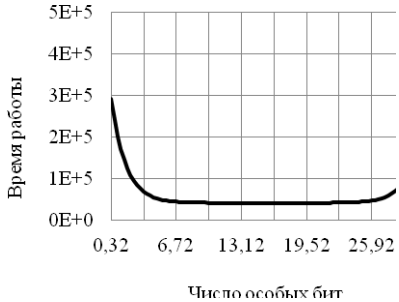

Рис. 2. Зависимость ожидаемого времени работы от числа особых бит в группе порядка 71

4. Экспериментальные результаты и выводы

Для реализации вышеуказанного алгоритма рассматривалась задача дискретного логарифма в группе точек эллиптической кривой над полем $GF(2^n)$. Был создан программный комплекс на языке C++ с использованием библиотек MPI (Message Passing Interface) и протестирован на вычислительном кластере. В результате удалось решить задачу в группе порядка 2^{71} на 96 процессах за 31,3 часа за $1{,}59 \cdot 10^8$ итераций при хранимом количестве особых точек $4{,}8 \cdot 10^5$ и параметре $\theta = 2^{-16}$. В работе [5] рассматривалась непараллельная реализация рассматриваемого метода на персональном компьютере. В данной работе авторам удалось решить задачу лишь в группах порядка не более 2^{53}.

Ниже приведены результаты тестирования программного комплекса в группе точек порядка 43 бит на 48 процессах при количестве памяти 1 Гб, позволяющей хранить в среднем $1{,}05 \cdot 10^6$

точек, в зависимости от выбранного параметра θ. Через I/T обозначено количество итераций в секунду — производительность алгоритма. При большом значении параметра θ наблюдается как увеличение количества итераций, так и уменьшение производительности, связанное с необходимостью пересылок и обработки большого количества данных.

На рисунке 3 представлена зависимость производительности алгоритма от количества особых бит s, а на рисунке 4 — зависимость времени работы программы от s.

Из рисунков 3 и 4 видно, что при малых значения параметра θ каждый из процессов способен генерировать почти максимальное количество точек, но максимум производительности не зависит от оптимального времени работы программы. Максимум достигается при $s \approx 14$. Полученные экспериментально данные вполне соответствуют результатам теоретических исследований.

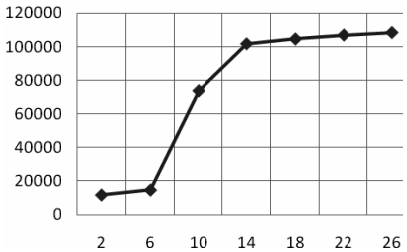

Рис. 3. Зависимость производительности от числа особых бит в группе порядка 43

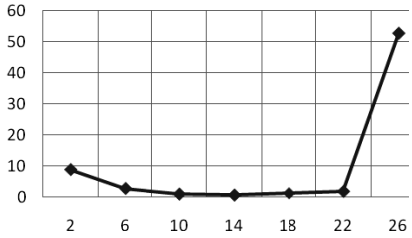

Рис. 4. Зависимость времени работы от числа особых бит в группе порядка 43

На рисунке 5 приведены результаты тестирования программного комплекса в группе точек порядка 43 бит при $\theta = 2^{-13}$ исходя из числа процессов p. Сплошная линия — среднее значение количества итераций I, пунктирная — ожидаемое их значение.

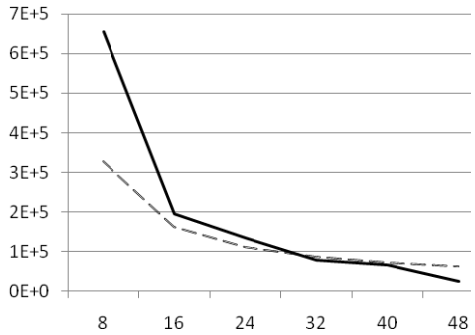

Рис. 5. Зависимость количества итераций от числа процессоров

Таким образом, можно сделать следующие выводы:

1) построена модель, определяющая оптимальные параметры параллельного ро-метода Полларда при его реализации в модели SPMD;

2) показано, что достаточно одного центрального процесса для хранения данных;

3) определен оптимальный способ пересылки точек центральному процессору;

4) создан комплекс программ для решения поставленной задачи, эффективность которого была протестирована на суперкомпьютере РГУ им. И. Канта;

5) произведен анализ данных, полученных экспериментально. Теоретические результаты были подтверждены.

Список литературы

1. *Van Oorschot P., Wiener M.* Parallel collision search with cryptanalytic applications // J. of Cryptology. 1999. 12(1).

2. *Hankerson D., Menezes A., Vanstone S.* Guide to elliptic curve cryptography. N.-Y., 2004.

3. *Kuhn F., Struik R.* Random walks revisited: extensions of Pollard's rho algorithm for computing multiple discrete logarithms // 8th Annual Workshop on Selected Areas in Cryptography. Toronto, 2001.

4. *Кнут Д. Э.* Искусство программирования. Т. 3: Сортировка и поиск. М., 2008.

5. *Перевощиков В. В., Гриценко А. А.* Об эффективной реализации дискретного логарифмирования на эллиптических кривых // Научная сессия ТУСУР. Т. 3: Системная интеграция и безопасность. Томск, 2009.

Об авторе

Виталий Витальевич Перевощиков — науч. сотр., Лейпцигский университет, Германия, e-mail: vitaly.perev@gmail.com

Author

Vitaliy Perevoshchikov — scientific collaborator, University of Leipzig, Germany, e-mail: vitaly.perev@gmail.com