Задание для параллельных вычислений

Реализация алгоритма дискретного логарифмирования Адлемана на GPU.

Необходимо реализовать алгоритм дискретного логарифмирования Адлемана с использованием GPU и на процессоре. Для реализации этого алгоритма нужно использовать следующие библиотеки для работы с видеокартой: OpenCL (C++), PyOpenCL (OpenCL для Python), Aparapi (OpenCL для Java), CUDA (C++). Приложение должно быть консольным. В итоге нужно сравнить скорость выполнения алгоритма на GPU и CPU.

Пример работы алгоритма:

1) Дана задача найти дискретный логарифм. $(\log_{21} 34 = x) \ mod \ 127$. По условию задача будет всегда иметь решение. (p = 127)

2) Найдем размер факторной базы: $B = e^{\sqrt{(\log_2 p) * (\log_2 (\log_2 p))}}$. В программе логарифм в данной формуле должен быть вычислен по основанию 2, но для примера возьмем вычисления по натуральному логарифму (для получения меньшего значения факторной базы, для вычисления вручную). $B \approx 84$ по основанию 2, а по натуральному логарифму $B \approx 16$ (его и будем использовать для дальнейших вычислений в примере).

3) Факторная база будет состоять из всех простых чисел $< B$: 2, 3, 5, 7, 11, 13

4) Далее необходимо найти следующие логарифмы:

$$\log_{21} 2 = ?$$
$$\log_{21} 3 = ?$$
$$\log_{21} 5 = ?$$
$$\log_{21} 7 = ?$$
$$\log_{21} 11 = ?$$
$$\log_{21} 13 = ?$$

5) Для нахождения логарифмов из шага 4 нужно возводить основание 21 в случайную степень по модулю 127. Если полученное число B гладкое (при его разложении на простые множители все его множители $< B$) оставляем его, в противном случае отбрасываем. Таким образом, случайно перебирая показатель степени набираем необходимое количество уравнений, объединяя их в СЛАУ. Данный шаг необходимо параллельно выполнять на видеокарте.

$21^4 = 44 \ (mod \ 127) = 2^2 * 11$

$21^5 = 35 \ (mod \ 127) = 7 * 5$

$21^6 = 100\ (mod\ 127) = 2^2 * 5^2$

$21^7 = 68\ (mod\ 127) = 2^2 * 17$ – не B гладкое отбрасываем

$21^8 = 31\ (mod\ 127)$ – не B гладкое отбрасываем

$21^9 = 2^4$

$21^{10} = 82\ (mod\ 127) = 2 * 41$ – не B гладкое отбрасываем

$21^{18} = 2$

$21^{25} = 3^2$

$21^{29} = 3 * 5$

$21^{31} = 11$

$21^{32} = 2^3 * 13$

На данном шаге перебор закончен, так как набраны необходимые уравнения для нахождения всех неизвестных.

6) Решаются уравнения следующим образом:

$$\begin{cases} 21^4 = 44\ (mod\ 127) = 2^2 * 11 \\ 21^{18} = 2 \end{cases}$$

Логарифмируем обе части уравнений:

$\log_{21} 21^4 = \log_{21}(2^2 * 11)\ mod\ (127 - 1)$. Модуль уменьшен на 1, так как переходим к мультипликативной группе, а в ней нет 0, следовательно, на 1 элемент меньше.

$$\begin{cases} 4 = 2\log_{21} 2 + \log_{21} 11\ \ (mod\ 126) \\ 18 = \log_{21} 2\ \ (mod\ 126) \end{cases}$$

$4 = 2 * 18 + \log_{21} 11\ \ (mod\ 126)$

$-32 = \log_{21} 11\ \ (mod\ 126)$

$\log_{21} 11 = 94\ (mod\ 126)$

И так далее…

Для вычисления на компьютере данные уравнения можно представить в виде коэффициентов матрицы СЛАУ.

1 0 0 0 0 0 | 18

0 2 0 0 0 0 | 25

0 0 1 1 0 0 | 29

0 0 0 0 1 0 | 31

3 0 0 0 0 1 | 32

Метод решения СЛАУ в кольцах вычетах (https://www.hse.ru/data/049/621/1235/003.pdf). При его реализации все операции с матрицами, а также вычисления НОД производить на GPU.

Можно предложить любой другой алгоритм решения СЛАУ при условии, что он вычисляет неизвестные во всех конечных кольцах вычетов и его шаги можно вычислять на GPU параллельно.

По итогу получаем все вычисленные логарифмы на 4ом шаге (система решалась вручную).

$$\log_{21} 2 = 18$$

$$\log_{21} 3 = \text{нет решения}$$

$$\log_{21} 5 = \text{нет решения}$$

$$\log_{21} 7 = \text{нет решения}$$

$$\log_{21} 11 = 94$$

$$\log_{21} 13 = 104$$

7) Получили таблицу факторной базы. На последнем шаге опять случайно выбираем число и решаем следующие выражение:

$34 * 21^{random}\ mod\ 127$, где 34 и 21 это значения из 1го шага

Пусть random = 2, тогда $34 * 21^2 = 8 = 2^3\ mod\ 127$

$\log_{21} 34 + 2 * \log_{21} 21 = 3 * \log_{21} 2\ mod\ 126$

$\log_{21} 34 = 52\ mod\ 126$

Ответ: 52