Семинар 2: Генераторы случайных чисел

Существуют 3 типа генераторов случайных чисел: аппаратный генератор случайных чисел и два программных типа генераторов - генератор квазислучайных чисел и генератор (псевдо)случайных чисел (ГСЧ). Мы остановимся на генераторах (псевдо)случайных чисел, наиболее общем типе детерминистических генераторов. В основе ГСЧ лежит строгий математический алгоритм, генерирующий последовательность случайных чисел u_i из единичного интервала [0,1). ГСЧ хорошего качества должен иметь длинный период и генерировать последовательность независимых равномерно и одинаково распределённых чисел, иммитирующую последовательность случайных величин. В молекулярной динамике в неявном растворителе и в динамике Ланжевена распределённые по Гауссу случайные числа используются для моделирования случайных столкновений с молекулами воды. Для генерации случайной силы последовательность равномерно распределённых случайных чисел u_i, преобразуется в последовательность нормально распределённых случайных чисел g_i при помощи специальных преобразований. В данной работе мы использовали наиболее популярное из таких преобразований - преобразование Бокса-Мюллера [1].

1 Реализация генераторов случайных числе на графическом процессоре

Для разработки параллельных реализаций ГСЧ на ГП можно использовать метод разделение цикла генератора [2]. Основная идея метода заключается в распределении одной последовательности ГСЧ, которая может быть рассмотрена как периодический цикл слу-

чайных чисел, между множеством потоков, каждый из которых производит подпоследовательность случайных чисел. Большинство ГСЧ, включая LCG, Ran2 и гибридный Таус, основаны на последовательном преобразовании текущего состояния, поэтому наиболее естественным способом получения разных последовательностей случайных чисел в разных потоках является сообщение потокам различных начальных значений состояния. При этом начальные состояния распределяются между потоками таким образом, чтобы исключить межпоточные корреляции случайных чисел. На этом основан подход “один-ГСЧ-на-поток” 1. Существуют ГСЧ, такие как вихрь Менсенна и алгоритм Фибоначчи, в которых можно перескочить вперёд по последовательности и вычислить случайное число $(n + 1)$, не вычисляя перед этим n-ное [2–4]. Длина запаздования, которая зависит от параметром ГСЧ, может быть выбрана больше числа потоков (которое равно числу частиц). Тогда все N чисел могут быть получены одновременно, то есть. j-тый поток вычисляет j-тое, $(j + N)$-тое, $(j + 2N)$-тое и так далее числа. Следует отметить, что в конце каждого шага по времени все потоки должны быть синхронизированы, чтобы обновить текущее состояние ГСЧ. Также, только одно состояние ГСЧ используется для всех потоков, каждый из которых обновляет только один элемент этого состояния. Такой подход можно назвать “один-ГСЧ-на-все-потоки” 1.

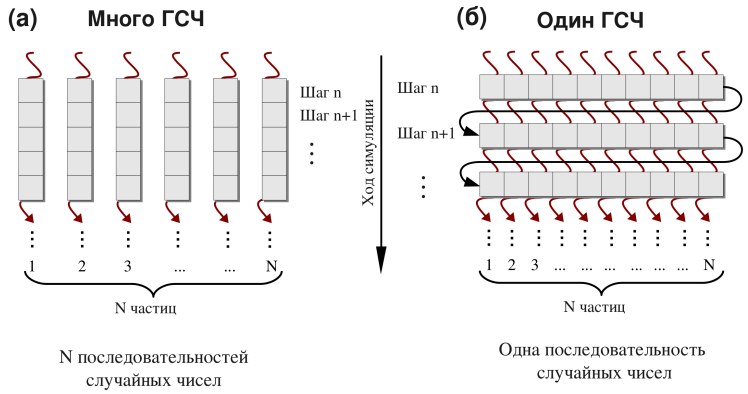

Рис. 1: Два подхода к реализации генераторов случайных чисел на ГП.

2 Алгоритмы

Алгоритм 1: Реализация алгоритма гибридный Таус на ГП.

Require: $y_1^h[N]$, $y_2^h[N]$, $y_3^h[N]$ и $y_4^h[N]$ выделены в памяти ЦП

Require: $y_1^d[N]$, $y_2^d[N]$, $y_3^d[N]$ и $y_4^d[N]$ выделены в глобальной памяти ГП

1. $y_1^h[1 \dots N]$ - $y_4^h[1 \dots N] \leftarrow$ начальное состояние
2. $y_1^h[1 \dots N]$ - $y_4^h[1 \dots N] \rightarrow y_1^d[1 \dots N]$ to $y_4^d[1 \dots N]$ {копирование начального состояния на ГП}
3. Начало кода для ГП
4. $j_{th} \leftarrow$ индекс потока
5. y_1, y_2, y_3 и $y_4 \leftarrow y_1^d[j_{th}]$, $y_2^d[j_{th}]$, $y_3^d[j_{th}]$ and $y_4^d[j_{th}]$ {загрузка состояния генератора}
6. **for** $i = 1$ to 4; $i++$ **do** {генерация случайных чисел}
7. $b \leftarrow (((y_1 \ll c_{11}) \text{ XOR } y_1) \gg c_{21})$
8. $y_1 \leftarrow (((y_1 \text{ AND } c_1) \ll c_{31}) \text{ XOR } b$
9. $b \leftarrow (((y_2 \ll c_{12}) \text{ XOR } y_2) \gg c_{22})$
10. $y_2 \leftarrow (((y_2 \text{ AND } c_2) \ll c_{32}) \text{ XOR } b$
11. $b \leftarrow (((y_3 \ll c_{13}) \text{ XOR } y_3) \gg c_{23})$
12. $y_3 \leftarrow (((y_3 \text{ AND } c_3) \ll c_{33}) \text{ XOR } b$
13. $y_4 \leftarrow ay_4 + c$
14. Вывод $mult \times (\text{ XOR } y_1 \text{ XOR } y_2 \text{ XOR } y_3 \text{ XOR } y_4)$
15. **end for**{генерация следующего случайного числа}
16. y_1, y_2, y_3 и $y_4 \rightarrow y_1^d[j_{th}]$, $y_2^d[j_{th}]$, $y_3^d[j_{th}]$ и $y_4^d[j_{th}]$ {сохранение текущего состояния генератора}
17. Конец кода для ГП

В приведенном листинге b - временная целочисленная беззнаковая переменная, y_1, y_2, y_3, y_4 - беззнаковые целые переменные состояния для трех генераторов Таусворта (строки 6−11) и одного LCG (строка 12), XOR - бинарное исключающее ИЛИ, "$\gg$" и "$\ll$" - бинарный сдвиг направо и налево, соответственно, $mult$=2.3283064365387$\times 10^{-10}$ - множи-

тель, который преобразует целое число в число с плавающей точкой от 0 до 1, c_{11}=13, c_{21}=19, c_{31}=12, c_{21}=2, c_{22}=25, c_{23}=4, c_{31}=3, c_{32}=11, c_{33}=17, c_1=4294967294, c_2=4294967288, c_3=4294967280 - постоянные параметры генераторов Таусворта [5], a=1664525 и c=1013904223 постоянные параметры LCG [6].

Алгоритм 3: Аддитивный алгоритм Фибоначчи с запаздыванием.

Require: $x^d[N]$ выделено в глобальной памяти ГП

1. $x^d[1 \dots ll] \leftarrow$ начальное состояние;
2. **for** $t = 0$ to S **do** {начало симуляций}
3. Начало кода для ГП
4. $j_{th} \leftarrow$ индекс потока
5. $shift_0 \leftarrow (j_{th} + N * t) * RNS$
6. **for** $shift = shift_0$ to $shift_0 + RNS - 1$ **do**
7. $x_{ll} \leftarrow x^d[shift \bmod ll]$
8. $x_{sl} \leftarrow x^d[(shift + sl - ll) \bmod ll]$
9. $x \leftarrow (x_{ll}\ op\ x_{sl}) \bmod m$
10. вывод x
11. $x \rightarrow x^d[shift \bmod ll]$
12. **end for**
13. Конец кода для ГП
14. **end for**

Чтобы инициализировать ГСЧ, ЦП помещает ll чисел начального состояния в массив x^d и копирует их в глобальную память ГП (строка 1). На ГП, каждый поток вычисляет положение ($shift_0$) числа, которое соответствует положению первой случайно величины, которую нужно сгенерировать. Это делается с использованием значения текущего шага интегрирования (t), номера потока (j_{th}), числа потоков (N) и количества случайных величин, требующихся на каждом шаге (RNS). Строки 6−10 повторяются до тех пор, пока не будет сгенерировано RNS случайных чисел (цикл начинается со строки 5), используя оператор op (строка 8). Для вычисления каждого случайного числа необходимо считать

два целых числа из состояния ГСЧ (строки 6 and 7). Эти числа соответствуют длинному запаздыванию ll и короткому запаздыванию sl. Все значения позиций в массиве состояния вычисляются по модулю ll, что соответствует "зацикливанию" массива состояния (числа начинают читаться/сохраняться в начало массива, когда достигнут его конец). Полученное целое x сохраняется для использования на следующих шагах интегрирования (строка 10). Когда требуется RNS случайных чисел в каждом потоке и на каждом шаге, то должно быть выполнено условие $sl{>}RNS{\times}N$ и $ll - sl{>}RNS{\times}N$, поскольку на каждом шаге обновляется $RNS{\times}N$ чисел состояния генератора.

Список литературы

[1] G. E. P. Box and M. E. Muller, "A note on the generation of normal random deviates," *Ann. Math. Stat.*, vol. 29, pp. 610–611, 1958.

[2] M. Mascagni and A. Srinivasan, "Parameterizing parallel multiplicative lagged-Fibonacci generators," *Parallel Comput.*, vol. 30, no. 7, pp. 899–916, 2004.

[3] G. Marsaglia, "Random numbers for C: The END?," 1999. Published on sci.crypt.

[4] P. L'Ecuyer, F. Blouin, and R. Couture, "A search for good multiple recursive random number generators," *ACM T. Model. Comput. S.*, vol. 3, no. 2, pp. 87–98, 1993.

[5] P. L'Ecuyer, "Maximally equidistributed combined tausworthe generators," *Math. Comput.*, vol. 65, no. 213, pp. 203–213, 1996.

[6] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C.* The Art of Scientific Computing, Cambridge University Press, second ed., 1992.