Генератори псевдовипадкових послідовностей з використанням обчислень на відеокартах

Пішта Я.В.
«Київський політехнічний інститут», pishta.yaroslav@bk.ru

The article analyzed the peculiarities of the *generators of random sequences, with using video cards. Also, research results and their analysing.*

Вступ

У наш час все більшого поширення набувають різні методи і алгоритми приблизних обчислень, так звані методи Монте-Карло. Це загальна назва групи чисельних методів, заснованих на отриманні великого числа реалізацій стохастичного (випадкового) процесу, який формується таким чином, щоб його імовірнісні характеристики співпадали з аналогічними величинами розв'язуваної задачі. Методи використовуються для вирішення завдань в різних галузях фізики, хімії, математики, економіки, оптимізації, теорії управління та ін.

Ці методи маю у своєму складі генератори випадкових чисел, і вимагають складних обчислень, і тому для реалізації їх вдаються до останніх досягнень науки і техніки.

Одним з таких досягнень - є розвиток графічних відеоадаптерів, і використання їх для обчислень. Але організація розподілених обчислень висуває рад вимог, до алгоритмів і методів, а способи їх вирішення потребують вивчення.

Постановка проблеми.

Методи Монте-Карло засновані на використанні генераторів псевдовипадкових послідовностей. І не в залежності від складності алгоритму - до генераторів висуваються радий вимог - великий цикл затримки, а саме головне «вагу» (мається на увазі необхідна пам'ять для функціонування генератора) бо необхідно мати на увазі обмеженість ресурсів відеокарти.

Мої дослідження спрямовані на вивчення різних методів генерації псевдовипадкових чисел, їх аналіз, і синтез методу який відповідатиме поставленим вимогам.

Термінологія

Генератор псевдовипадкових чисел (ГПСЧ, англ. Pseudorandom number generator, PRNG) - алгоритм, що генерує послідовність чисел, елементи якої майже незалежні один від одного і підкоряються заданому розподілу(зазвичай рівномірному).

CUDA (англ. Compute Unified Device Architecture) - програмно-апаратна архітектура, що дозволяє робити обчислення з використанням графічних процессоров NVIDIA, що підтримують технологію GPGPU (довільнихобчислень на відеокартах). Вперше з'явилися на ринку з виходом чіпа NVIDIA. Восьмого покоління - G80 і присутній у всіх наступних серіях графічних чіпів,які використовуються в родинах прискорювачів GeForce, Q uadro і NVidia Tesla.

Дослідження

У ході аналізу існуючих рішень були виявлені основні способи досягання поставленого результату. При дослідженні я сконцентрувався на підходах організації генерації псевдовипадкових чисел «генератор псевдовипадкових послідовностей-на один потік», і «генератор псевдовипадкових послідовностей – на всі потоки».

Підходи відрізняються тим що в першому випадку кожен потік «забезпечується» власним генератор псевдовипадкових послідовностей1), другий же алгоритми увазі, що його змінні (значення) будуть використовуватися всіма потоками одночасно, там мінімізується повторюваність значень у кількох потоках.

Кожен генератор псевдовипадкових послідовностей видає послідовність випадкових чисел ui, які повинні являти собою незалежні значення в інтервалі [0,1].

Можна виділити основні 3 вимоги до програмних реалізацій генераторів псевдовипадкових послідовності[1]:

- Хороші статистичні характеристики
- Висока швидкість обчислень
- Мале споживання пам'яті.

Оскільки детермінована послідовність випадкових числі в кінцевому рахунку приходить у вихідну точку,

$$U(n + p) = Un \qquad (1)$$

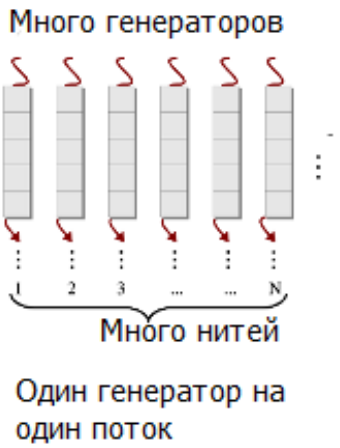
у генератора псевдовипадкових послідовностей повинен бути великий період. А також генератор повинен пройти строгі тести на незалежність і рівномірність.

Існує досить багато методів отримання нормального Розподілення випадкових чисел. В ході дослідження була проведена адаптація методу перетворення Бокса-Мюллера[2].

У підході «Один генератор випадкових числі на потік» ідея полягає в тому, що кожен потік, що виконується паралельно організований таким чином, щоб генерувати випадкові послідовності не залежно від інших[3]. Так велика частина алгоритмів генерації (Mersenne Twister і Lagged Fibonacci), які використовують рекурсивне перетворення дозволяють отримати (n +1)-е число не знаючи n-го числа. Кількість одержуваних таким методом чисел, яке, в загальному випадку, залежить від вибору параметрів для генераторів псевдовипадкових послідовностей, має дорівнювати числу потоків N або кратному йому числу M × N. Тоді всі випадкові числа можна отримати одночасно, тобто j-а нитка отримує числа

$$j, j + N, j + 2N, \ldots \qquad (2)$$

Але в кінці кожного кроку симуляції нитки необхідно синхронізувати, щоб оновити поточний стан генератора випадкових чисел. В результаті такого підходу однаковий стан генератора випадкових чисел може бути використано всіма потоками, при цьому оновлення буде проходіть тільки одного елемента стану (рис. 1).



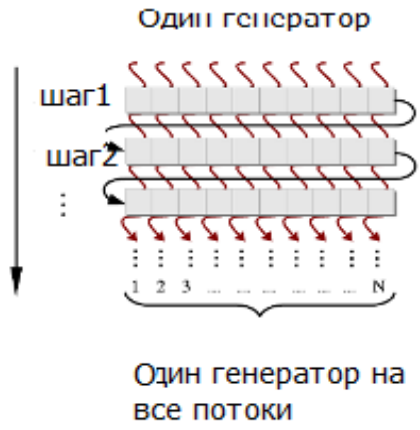


Рисунок 1 - Типи організації генераторів

Ці методи були використані для реалізації лінійного конгруентного генератора (LCG) і Ran2, Hybrid Taus і алгоритму Lagged Fibonacci (рис.2).

Значне підвищення буде спостерігатися при генерації псевдовипадкових чисел на відеокарті.

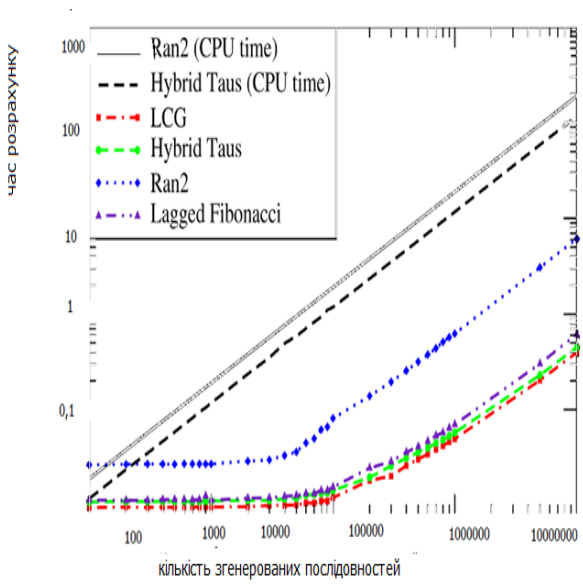


Рисунок 2 - Відношення часу обчислення на GPU ЦП випадкових послідовностей

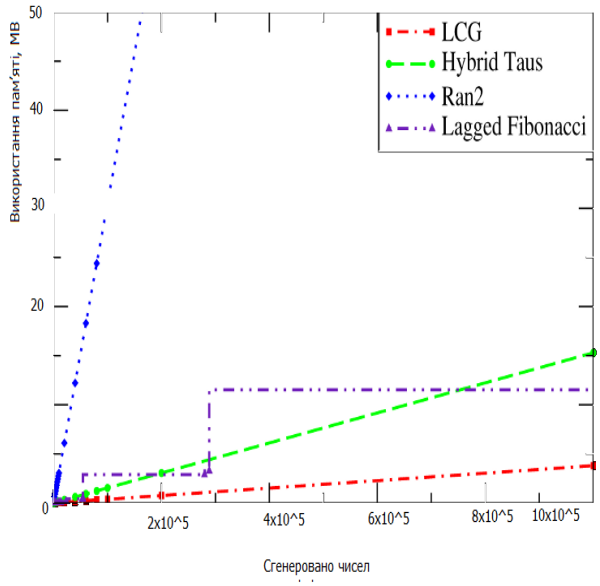


Рисунок 3. Порівняння обчислювальної ефективності

Результати, отримані для 2,72 ГГц ЦП Intel Core i3 230 і GeForce 320 показують, що при певній кількості згенерованних випадкових чисел стає невигідно їх генерувати на ЦП, та передавати на відео карту.

Під час дослідження були проведені порівняння обчислювальної ефективності генераторів псевдовипадкових послідовностей, результати показані на рис.3.

Висновки

Результати досліджень показали, що різні генератори значною мірою відрізняються за споживаним ресурсів.

При цьому найбільш ефективнимвиявився алгоритм генерації псевдовипадкових послідовностей на основілінійного конгруэнтного генератора.

Література

[1] Zhmurov A, Dima RI, Kholodov Y, Barsegov V. SOP-GPU: Accelerating biomolecular simulations in the centisecond timescale using graphics processors. Proteins 2010; 78: 2984–2999.

Шнайер Б. 2.8 Генерация случайных и псевдослучайных последовательностей // Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си = Applied Cryptography. Protocols, Algorithms and Source Code in C — М.: Триумф, 2002. — 816 с

Gilks, W. R.; Richardson, S.; and Spiegelhalter, D. J. (Eds.). Markov Chain Monte Carlo in Practice. Boca Raton, FL: Chapman & Hall, 1996.