Использование суперкомпьютеров для решения некоторых задач криптоанализа.

А.И. Миненко
магистрант СибГУТИ г. Новосибирск, Россия
minenkoai@mail.ru

Введение

Одной из задач криптоанализа является тестирование генераторов случайных чисел. Во многих протоколах и шифрах случайные слова и числа используются как секретные ключи.
Более того, с развитием криптографии выяснилось, что многие фундаментальные проблемы этой науки тесно связаны с генерированием и тестированием случайных чисел. Генераторы случайных чисел также находят широкое применение в вычислительных методах и при имитационном моделировании.

Эта задача привлекает внимание многих исследователей в силу её теоретической и практической важности. Так, Национальный институт стандартов США (NIST) разработал пакет статистических тестов для проверки бинарных последовательностей, созданных либо аппаратными, либо программными генераторами случайных чисел.

Кроме этих тестов, существуют и другие. Одним из них является статистический тест “Стопка книг”, который был предложен Рябко Б. Я.

Введение (продолжение)

Одно из требований, предъявляемое к современным генераторам: генерируемая случайная последовательность должна быть статистически неотличима от абсолютно случайной.

Методы, используемые для проверки этого условия, рассматриваются в рамках математической статистики. Таким образом, необходимо проверить гипотезу H_0 о том, что источник порождает символы алфавита равновероятно и независимо, против альтернативной гипотезы H_1, являющейся отрицанием H_0.

Применение суперкомпьютеров

Рассматриваемая задача имеет высокую трудоёмкость вычислений, так как каждым тестом нужно проверить несколько последовательностей, которые могут быть очень большими.

Каждая из последовательностей может быть проанализирована на отдельном вычислителе, поэтому для решения этой задачи хорошо подходит суперкомпьютер. Кроме того, распараллелить можно выполнение самих тестов.

Стопка книг

Основная идея метода в упорядочивании элементов как в стопке книг. Книга вынимается из стопки и кладётся наверх. Её номер становиться первым; книги, которые до этого были над ней сдвигаются вниз, а остальные остаются на месте. Дадим краткое описание этого теста.

Пусть некоторый источник порождает буквы из алфавита $A = \{a_1, a_2, \ldots, a_s\}$, $S > 1$, и требуется по выборке $x_1, x_2, ..., x_n$ проверить гипотезу

$$H_0 : p(a_1) = p(a_2) = ... = p(a_s) = 1/S \tag{1}$$

против альтернативной гипотезы H_1, являющейся отрицанием H_0.

При тестировании по предлагаемому методу буквы алфавита A упорядочены (и занумерованы в соответствии с этим порядком от 1 до S), причём этот порядок меняется после анализа каждого выборочного значения как в стопке книг. Обозначим через $v^t(a)$ номер буквы $a \in A$.

Стопка книг (продолжение)

При применении описываемого теста множество всех номеров $\{1, ..., S\}$ заранее, до анализа выборки, разбивается на $r > 1$ непересекающихся частей $A_1 = \{1, 2, ..., k_1\}$, $A_2 = \{k_1 + 1, ..., k_2\}$, ..., $A_r = \{k_{r-1} + 1, ..., k_r\}$.

Затем по выборке $x_1, x_2, ..., x_n$ подсчитывается количество номеров $v^t(x_t)$, принадлежащих подмножеству A_j, которое мы обозначим через n_j, $j = \overline{1, r}$.

При выполнении гипотезы H_0 вероятность того, что номер принадлежит множеству A_j, пропорциональна количеству элементов в этом подмножестве, т.е. равна $|A_j|/S$. Затем по критерию χ^2 проверяется гипотеза

$$H_0^* : P\{v^t(x_t) \in A_j\} = |A_j|/S \tag{2}$$

против альтернативной гипотезы $H_1^* = \neg H_0^*$. Очевидно, при выполнении исходной гипотезы H_0 выполняется H_0^*, и наоборот, при выполнении гипотезы H_1^* выполняется H_1.

Тесты NIST

В состав пакета NIST входят 15 статистических тестов. Перечислим их:

1) Frequency (Monobit)
2) Frequency within a Block
3) Runs
4) For the Longest Run of Ones in a Block
5) Binary Matrix Rank
6) Discrete Fourier Transform (Spectral)
7) Non-overlapping Template Matching
8) Overlapping Template Matching
9) Maurer's "Universal Statistical"
10) Linear Complexity
11) Serial
12) Approximate Entropy
13) Cumulative Sums (Cusum)
14) Random Excursions
15) Random Excursions Variant

Эти тесты основаны на различных особенностях, присущих только неслучайным последовательностям. Во всех тестах, если вычисленное в ходе теста значение P-value $<$ 0,01, то данная двоичная последовательность не является истинно случайной. В противном случае, последовательность носит случайный характер.

Генераторы

В этом исследовании протестированы 18 линейных конгруэнтных генераторов, RC4 и функция rand() в компиляторе C++ gcc 4.3.2 Linux.

Линейные конгруэнтные генераторы вычисляются по формуле

$$X_{n+1} = (aX_n + c)\ mod\ m, \tag{3}$$

где a, c, m, X_0 - параметры метода.
Будем обозначать эти генераторы LCG(m, a, c). Параметр $X_0 = 1$ для всех генераторов. Приведём список линейных генераторов, которые протестированы:

1) LCG(2^{24}, 16598013, 12820163), этот генератор используется в Microsoft VisualBasic 6.0.
2) LCG(2^{31}, 65539, 0), RANDU долгое время использовался во многих компьютерах в 1960-х - 1970-х годах.
3) LCG(2^{32}, 1099087573, 0), этот генератор предложил Fishman.
4) LCG(2^{32}, 69069, 1), этот генератор предложил Marsaglia.
5) LCG(2^{32}, 69069, 5) использовался в компиляторах GNU.
6) LCG(2^{32}, 1664525, 1013904223) предложен в Numerical Recipes.
7) LCG(2^{32}, 22695477, 1) используется в Borland C/C++.
8) LCG(2^{32}, 1103515245, 12345) используется в Digital Mars.

Генераторы (продолжение)

9) LCG(2^{32}, 134775813, 1) используется в Borland Delphi.
10) LCG(2^{32}, 214013, 2531011) используется в Microsoft Visual/Quick C/C++.
11) LCG(2^{46}, 5^{13}, 0) использовался для аэродинамического моделирования в НАСА в исследовательском центре Ames.
12) LCG(2^{48}, 25214903917, 11) это генератор drand48 из стандартной библиотеки Unix.
13) LCG(2^{48}, 5^{19}, 0) это традиционный генератор, использующийся в Национальной лаборатории США Los Alamos.
14) LCG(2^{48}, 33952834046453, 0) это один из генераторов, который предложил Fishman.
15) LCG(2^{48}, 44485709377909, 0) использовался в системе CRAY.
16) LCG(2^{59}, 13^{13}, 0) базовый генератор в математической библиотеке NAG, также он присутствует в векторной статистической библиотеке(VSL), которая находится в библиотеке математического ядра Intel.
17), 18) LCG(2^{63}, 5^{19}, 1), LCG(2^{63}, 9219741426499971445, 1) рекомендовано использовать на будущее в Национальной лаборатории США Los Alamos.

Описание экспериментов

Каждый генератор выдавал 100 последовательностей одинаковой длины. В среднем 1 последовательность из 100 может быть забракована при уровне значимости $\alpha = 0.01$.
Доверительный интервал, вычисленный с помощью критерия χ^2, равен [0;4].
Если количество забракованных последовательностей не попадает в этот интервал, то это говорит о том, что генератор выдаёт последовательности, при данной длине выборки, статистически отличимые от случайных. В этом случае будем говорить, что генератор не прошёл испытания, то есть забракован.

Тестирования проводились для последовательностей, выдаваемых генераторами, от 2^8 до 2^{23} бит. Для некоторых генераторов тестирование было проведено при больших длинах последовательности и только некоторыми тестами.

Тестом "Стопка книг" последовательность $x_n \in \{0, 1\}$ разбивалась на блоки длины l и при тестировании рассматривалась как выборка из алфавита размера $S = 2^l$. Множество всех позиций в “Стопке книг” разбивалось на два подмножества $A_1 = \{a_1, a_2, ..., a_k\}$, $A_2 = \{a_{k+1}, ..., a_S\}$. Второе подмножество не хранилось в памяти компьютера.

При исследовании тестами NIST, выбирались рекомендуемые параметры.

Тестирование генераторов

Генератор/Тест	$Book\ \ stack$	$1.Frequency$	$2.Frequency\ \ Block$	$3.Runs$	$4.Longest\ \ Run$	$5.BinaryMatrix$	$6.DFT$	$7.Non-overlapping$	$8.Overlapping$	$9."Universal"$	$10.Linear\ \ Complexity$	$11.Serial$	$12.Approx\ \ Entropy$	$13.Cumulat\ \ Sums$	$14.Random\ \ Excurs$	$15.Random\ \ Exc\ \ Var$
$LCG(2^{24}, 16598013, 12820163)$	2^{16}						2^{21}					2^{23}				
$LCG(2^{31}, 65539, 0)$	2^{13}						2^{22}					2^{20}				
$LCG(2^{32}, 1099087573, 0)$	2^{20}						2^{23}					2^{23}				
$LCG(2^{32}, 69069, 1)$	2^{20}															
$LCG(2^{32}, 69069, 5)$	2^{20}															
$LCG(2^{32}, 1664525, 1013904223)$	2^{23}						2^{23}									
$LCG(2^{32}, 22695477, 1)$	2^{20}															
$LCG(2^{32}, 1103515245, 12345)$	2^{23}															
$LCG(2^{32}, 134775813, 1)$	2^{20}															
$LCG(2^{32}, 214013, 2531011)$	2^{19}															
$LCG(2^{46}, 5^{13}, 0)$																

Тестирование генераторов (продолжение)

Генератор/Тест	*Book stack*	1.*Frequency*	2.*Frequency Block*	3.*Runs*	4.*Longest Run*	5.*BinaryMatrix*	6.*DFT*	7.*Non − overlapping*	8.*Overlapping*	9."*Universal*"	10.*Linear Complexity*	11.*Serial*	12.*Approx Entropy*	13.*Cumulat Sums*	14.*Random Excurs*	15.*Random Exc Var*
$LCG(2^{48}, 25214903917, 11)$																
$LCG(2^{48}, 5^{19}, 0)$																
$LCG(2^{48}, 33952834046453, 0)$																
$LCG(2^{48}, 44485709377909, 0)$																
$LCG(2^{59}, 13^{13}, 0)$																
$LCG(2^{63}, 5^{19}, 1)$																
$LCG(2^{63}, 9219741426499971445, 1)$																
$RC4$	2^{32}															
$rand$ ($C++$ gcc 4.3.2)																

Вывод

Проведённые исследования позволяют дать следующие выводы и рекомендации по применению рассмотренных тестов и генераторов.

1) Тест "Стопка книг" может эффективнее находить отклонения от случайности, чем другие тесты, так как он нашёл отклонения на большем количестве генераторов. Во многих случаях это сделано при меньшей длине последовательности. А в некоторых случаях отклонения обнаруживает только "Стопка книг".

2) Линейные генераторы с параметром $m \leq 2^{32}$ не рекомендуется использовать в современных приложениях. RC4 рекомендуется использовать для создания последовательностей длиной до 2^{32} бит. Остальные генераторы можно использовать, так как до 2^{23} не было найдено отклонений выше перечисленными тестами.

Рекомендуемые параметры для “Стопки книг”: l - длину блока рекомендуется выбирать из ряда 8, 16, 20, 24, 32, 40 и так далее; размер одного или нескольких подмножеств $|A_i| = b * \sqrt{2^l}$, где b число из ряда 2, 4, 5, 8, 10, 16, 20, 32, 40, 64, 80, 128, 160 и так далее.