Randest - a randomness tests collection

Христо Стоянов

1 декември 2013

> “ Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin.
>
> John von Neumann ”

Абстракт

Използването на случайни числа намира широка употреба в множество статистически и алгоритмични задачи. След възникналата нужда се създават различни начини за генериране на редица от произволни числа - хвърляне на зар, теглене на топче от торба, раздаване на карти. В средата на XX век се създават първите устройства с такава цел. За оценяването на получените редици се описват множество статистически тестове, които се стремят да открият вътрешна зависимост. Randest е софтуерна библиотека от стохастични тестове, имплементирани на C++. Тестовете за това дали редица от числа е случайна са генерализирани и разчитат на единствено на определени свойства на типа данни. Създадената рамка за имплементация на тестове спомага за лесното разширяване с нова функционалност.

Абстракт

Random number have widespread usage in a variety of statistical and algorithmic tasks. Through history different ways for generating arbitrary number have been used - roll a dice, draw balls out of a “well-stirred urn,” or dealing out cards. The first devices with this goal were created in the middle of the XX century. A number of stochastic tests have been devised in order to try finding an underlying dependency. Randest is a software library of such tests, implemented in C++. Randomness tests can be applied on various types of data, as long as certain operators are defined. The framework allows for easy addition of new tests.

1 Въведение

Числа, които са били „избрани“ случайно, намират широко приложение в разнообразни области. Нуждата от тях може да се наблюдава например при симулация на различни процеси. Така наречените рандомизирани алгоритми често постигат по-добро представяне спрямо детерминистичните си еквиваленти.

> “ One of us recalls producing a “random” plot with only 11 planes, and being told by his computer center’s programming consultant that he had misused the random number generator: “We guarantee that each number is random individually, but we don’t guarantee that more than one of them is random.” Figure that out. ”
>
> W. H. Press

Едно число само по себе си няма как да определим дали е случайно. Нима числото 2 не е вероятен избор? Ами 11111? От друга страна, както става ясно в горния цитат, създаването на редица от числа, които да са правилно разпределени в определено разпределение е доста по-трудоемка задача. [8]

Възниква следният проблем - компютърът, една напълно детемирнистична машина, трябва да създава призволна, съответно непредвидима, редица от числа. Оставайки настрана този „грях“, описан така от Нойман, и за удобство, ще наричаме такива редици псевдо-случайни и ще формулираме свойствата, които те трябва да имат.

Поради липсата на по-добра дефиниция за редица от псевдо-случайни числа ще използваме следната: [4]

Дефиниция 1. Числата в една редица са произволни, ако никой от тестове, които прилагаме върху нея, не успява да отхвърли това твърдение.

Ако някой от прилаганите тестове показва голямо отклонение от теоретичното разпределение, то това означава, че спрямо тази характеристика разглежданата редица не е случайна. Предвид че обикновено има нужда от математически апарат, който да създава редици от привидно произволни числа, за една такава конструкция, ще съдим по това дали успяваме да различим нейния продукт от теоретичното разпределение.

2 Мотивация

Криптографията е област, в която цели системи, широко използвани в съвременните банкови системи, валути и комуникации, включват като

градивен елемент възможността за избиране на призволно число. [9] От критично значение е този избор да бъде сигурен.

Пропуск в този градивен елемент може да доведе до пълното разбиване на сигурността на една система, както е показано в [2].

За изследването на различни конструкции създаващи редица от случайни числа, така наречените Pseudorandom Number Generators (PRNGs), се създават различни софтуерни библиотеки. Началото поставя Кнут през 1969 година. Една от най-новите такива библиотека се нарича TestU01, създадена през 2007. [5]

3 Принос

Статистическите тестове са стандартни, добре описани в множество източници. [4] [5] Някои от тях имат имплементации на C.

Използването на C++11 позволява някои от тестовете да се обобщят, доближавайки се до математическата дефиниция максимално. Така се дава възможност на база дефинирани няколко оператора за тип той да се използва за пресмятанията. Получената имплементация може да бъде преизползвана, в резултат на което лесно се създават нови дефиниции на тестове.

Пример за преизползване е употребата на теста на Колмогоров-Смирнов. В максималния тест се използва получената генерализирана имплементация, като $F(x)$ се явява параметър. Подробно описание има в §4.1.

При проектирането на библиотеката е постигната абстракция на източникът на данни. Това позволява тестовете да бъдат приложими независимисимо от това дали данните са записани във файл, генерират се динамично от PRNG или се доставят по друг канал. В допълнение, на базата на тази абстракция, наречена data provider, могат да се имплементират различни модификации на данните, които да запазват старите данни. Подреждането на данните в нарастващ ред подобрява скоростта на пресмятане на някои от статистическите характеристики, както е показано в §4.1. Имплементирайки източници на данни, които се явяват модификация на вече съществуващи, спомага за поддържането на добра абстракция между тестове и входа им.

Стриктното придържане към стандартите на използвания език (препоръчват се опции при компилиране -std=c++11 -pedantic -Werror -Wall) спомага за това библиотеката да остане лесно преносима на разнообразни архитектури. Наличието на някои възможности от стандартната библиотека на C++ е задължително условие.

Създадените абстракции позволяват лесното добавяне на нови тестове и формирането на множество от тестове.

4 Статистически характеристики

В тази секция ще се опишат тестовете, които са имплементирани към момента в Randest. Колмогоров-Смирнов и χ^2 тестът на Пиърсън са непараметрични статистически тестове, които определят до колко наблюдавано множество се различава от теоретичното разпределение.

Важно е да се отбележи, че никоя от описаните характеристики и тестове не дават еднозначен отговор дали разглежданата редица от числа е произволна. За потвърждаване или отхвърляне на тази хипотеза трябва да се разгледа статистическата значимост на резултатите от даден тест.

4.1 Колмогоров-Смирнов

Ако искаме да разгледаме разпределението на случайна стойност X, можем да го направим чрез функция на разпределението $F(x)$, така че:

$$F(x) = Pr(X \leq x) = \text{вероятността } (X \leq x).$$

Статистиката на Колмогоров-Смирнов се дефинира по следния начин [3]:

$$D_n = \sup_x |F_n(x) - F(x)|$$

F_n емпирична функция на разпределението за n независими и идентично разпределени наблюдения O_i се дефинира като:

$$F_n(x) = \frac{1}{n} \sum_{i=1}^{n} I_{O_i \leq x}$$

$$I_{O_i \leq x} = \begin{cases} 1, & O_i \leq x \\ 0, & O_i > x \end{cases}$$

Този тест ще бъде използван с $F(x) = x$, ако не е указано друго, като $x \in [0, 1]$.

4.2 χ^2 тест на Пиърсън

χ^2 тестът на Пиърсън се използва за разпознаване дали честотното разпределение на множество наблюдения се различава от очакваното

теоретично разпределение. В същността си представлява сборът от квадратите на разликата между наблюдаваната честота и очакваната честота за всяко възможно наблюдение. [7]

По-нататък в описаните тестове ще се използва очакване за равномерно разпределение на наблюденията. Очакването ще записваме така, като N е броят наблюдения, с които боравим, а n - броят различни възможни наблюдения:

$$E_i = \frac{N}{n}$$

С O_i ще означим честотата на срещане на наблюдението i в редицата, която изследваме. Тогава статистиката се дефинира като:

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

5 Тестове

5.1 Честотни тестове

Имплементирани са два варианта на честотния тест по отношение на цели числа. Първият разглежда дали наблюдаваните числа за равномерно разпределени. Вторият разглежда до колко наблюденията са равномерно разпределени в даден интервал.

Различаването между двата теста е необходимо. Съществуват генератори на произволни числа, чийто резултат е равномерно разпределен в множеството на възможните изходи, но то не е от последователни цели числа. ването между двата теста е необходимо. Съществуват генератори на произволни числа, чийто резултат е равномерно разпределен в множеството на възможните изходи, но то не е от последователни цели числа. ването между двата теста е необходимо. Съществуват генератори на произволни числа, чийто резултат е равномерно разпределен в множеството на възможните изходи, но то не е от последователни цели числа. ването между двата теста е необходимо. Съществуват генератори на произволни числа, чийто резултат е равномерно разпределен в множеството на възможните изходи, но то не е от последователни цели числа.

Пример за такъв генератор е Blum-Blum-Shub, който разглежда квадратични остатъци по модул. Свойствата му обуславят успешно приложение в криптографията, но пресмятането му е бавно за симулационни цели (детайли в [1]).

5.2 Максимален тест

Максималният тест е описан в добре позната литература и се счита за един от основните, които трябва да бъдат покрити, за по-нататъшното разглеждане на съответния генератор. [4] В същността си представлява вземане на максимален елемент от t поредни и разглеждането на статистиката на Колмогоров-Смирнов на получената нова редица.

За $0 \leq j < n$, нека

$$V_j = max(U_{tj}, U_{tj+1}, \ldots, U_{tj+t-1})$$

За получената редица V_j се разглежда Колмогоров-Смирнов с $F(x) = x^t$, $0 \leq x \leq 1$. [4]

За да потвърдим, че $F(x) = x^t$ е правилната функция на разпределението, ще разгледаме вероятността $max(U_1, U_2, \ldots, U_t) \leq x$. Тя е равна на вероятността $U_1 \leq x$, $U_2 \leq x$ и $\ldots$ и $U_t \leq x$, което е произведението на всяка от вероятностите, x^t.

5.3 Събирач на купони

Разглеждаме редицата от неотрицателни числа $X_0, X_1, \ldots$ with $0 \leq X_i < d$. Наблюдаваме дължините на последователни, нямащи общи членове, подредици, които са с минимална дължина, и съдържат всички неотрицателни числа от 0 до $d-1$. Можем да разглеждаме този тест като момче, което събира стикери от кроасани. Съществуват d вида стикери, които са случайно разпределени в опаковките. То трябва да продължава да яде кроасани, докато не събере поне един стикер от всеки вид.

Тестът взима под внимание параметър t, който указва максималната дължина, над която се приема, че дължината на тази подредица е t. От даден източник на данни тестът работи докато съществуват още данни или докато n пъти бъдат събрани всички купони. За простота се разглеждат единствено завършени колекции. [4]

Прилага се χ^2 тест върху получените бройки за всяка дължина $d, d+1, \ldots, t$.

5.4 Монотонен тест

Една редица редица може да бъде разглеждана спрямо това дали намалява или нараства. Дължината на такива монотонни подредици е характеристиката, която се разглежда в този тест. Стандартна практика е било да се разглеждат едновременно намаляващите и нарастващите подредици, но по-късно е показано, че употребата на χ^2 тест е грешна. [6]

Това се дължи на факта, че дължината на текущата монотонна редица зависи от дължината на предишната такава.

В библиотеката е имплементирана вариация на този тест, която Кнут характеризира като „по-проста и практична версия“. След като една монотонна редица бъде прекъсната, следващият елемент се пропуска и се започва разглеждането на нова такава. Резултатните дължини са независими една от друга и върху тях може да бъдет приложен χ^2 тест. [4]

Препоръчително е отделното разглеждане на намаляващи подредици и растящи подредици, поради което е имплементирана възможност за използването и на двете.

6 Бъдещо развитие

Основен елемент в бъдещето развитие на библиотеката се явява имплементирането на допълнителни тестове. Разширяването на възможностите ще доведе до създаването на библиотека, която да служи за пълноценен емпиричен анализ на различни източници от данни - наблюдения на процеси, генератори на псевдослучайни числа и други.

Формирането на класове тестове е следващата стъпка, когато бъдат имплементирани голямо множество такива.

Планирано допълнение към библиотеката е включването на генератори на случайни числа и разширяване на възможните източници на данни. Това ще спомогне за възможността да се сравняват статистическите резултати от различни генератори.

Интересна възможност, която заслужава внимание, е идеята да се създаде алгоритъм, който да адаптира параметрите на множество тестове, в зависимост от резултатите им. Предимството на такъв подход е автоматизацията. Към момента, при емпирично изследване, параметрите се изменят от човек, независимо каква библиотека от тестове се използва.

Литература

[1] L Blum, M Blum, and M Shub. A simple unpredictable pseudo random number generator. SIAM J. Comput., 15(2):364–383, May 1986.

[2] Flavio D. Garcia, Gerhard Koning Gans, Ruben Muijrers, Peter Rossum, Roel Verdult, Ronny Wichers Schreur, and Bart Jacobs. Dismantling mifare classic. In Proceedings of the 13th European Symposium on Research in Computer Security: Computer Security, ESORICS '08, pages 97–114, Berlin, Heidelberg, 2008. Springer-Verlag.

[3] Michiel Hazewinkel. Encyclopaedia of mathematics : an updated and annotated translation of the Soviet "Mathematical encyclopaedia". Reidel Sold and distributed in the U.S.A. and Canada by Kluwer Academic Publishers, Dordrecht Boston Norwell, MA, U.S.A, 1988. Details at http://www.encyclopediaofmath.org/index.php/Kolmogorov-Smirnov_test.

[4] D. E. Knuth. The Art of Computer Programming. Volume 2: Seminumerical Algorithms. Addison-Wesley, 1969.

[5] Pierre L'Ecuyer and Richard Simard. Testu01: A c library for empirical testing of random number generators. ACM Trans. Math. Softw., 33(4), August 2007. Повече информация можете да намерите на страницата на TestU01 http://www.iro.umontreal.ca/~simardr/testu01/tu01.html.

[6] H Levene and J Wolfowitz. The covariance matrix of runs up and down. The Annals of Mathematical Statistics, 15(1):58–69, 1944.

[7] Karl Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that can be reasonably supposed to have arisen from random sampling. Philosophical Magazine, 50:157–175, 1900.

[8] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. Numerical Recipes in C: The Art of Scientific Computing. Second Edition, 1992. Интерес представлява глава 7 - Случайни числа.

[9] Bruce Schneier. Applied Cryptography (2Nd Ed.): Protocols, Algorithms, and Source Code in C. John Wiley & Sons, Inc., New York, NY, USA, 1995.