松本眞さんの疑似乱数発生法について

二宮祥一 (東京工業大学理財工学研究センター)

松本眞さんが船井情報科学振興賞を受賞　広島大学大学院理学研究科教授の松本眞さんが第 4 回船井情報科学振興賞を受賞されました。財団法人船井情報科学振興財団の WEB ページから引用すると、この賞は「情報技術に関する研究について顕著な功績のあった研究者に対」して送られるものです。松本さんの「擬似乱数発生法の開発と評価法の研究及び普及活動」は「コンピューターサイエンス部門」で褒賞されました。松本さんはこれで Institute of Combinatorics and its Applications: Kirkman Medal(1997 年)、日本数学会建部賢弘賞 (1998 年)、慶應大学義塾賞 (1998 年)、日本 IBM 科学賞 (1999 年) に続きまた新たなトロフィーを得たことになります。

松本さんの研究分野は非常に広く、上述の様に沢山の賞を受けていることにもそれを見ることができます。これらの賞のうち、日本 IBM 科学賞と今回の船井情報科学振興賞はコンピュータサイエンスに関わる業績に対するもので、両者ともその驚異的な性能で知られる疑似乱数 “Mersenne Twister”(以下 MT と略します) の発明と疑似乱数に関わる研究に対して送られました。ここではこれらの業績、MT の開発と新しい非統計的検定手法の開発とについて説明させて頂きます。

MT 以前の疑似乱数の歴史　疑似乱数というのは、文字通り乱数に似たものをコンピュータ上で決定的なアルゴリズムに依り生成する方法やプログラムのことです。このプログラムが生成する数列がサイコロを振って生成された数列と区別が付かないことが理想ですが、これは原理的に不可能なので、如何にして「乱数っぽい」ものをつくるか、が目標になります。

Ulam が Monte Carlo 法のアイディアを思いつき、von Neumann との議論のなかでこれを発明したのはコンピュータの誕生の直後でした [8]。Monte Carlo 法にとって疑似乱数は最も重要なところですからこの時に疑似乱数の歴史も始まりました。つまり、疑似乱数はコンピュータとほぼ同時に生れたことになります。最初に良く用いられたのは von Neumann の開発した疑似乱数だったそうです。その後のコンピュータの急激な進歩に伴ない Monte Carlo 法の利用も広がり、疑似乱数の重要性は大きくなっていきました。Monte Carlo 法を行なうときに疑似乱数の質が計算結果に深刻な問題をもたらすということがわかってきたからです。ここでは主として Monte Carlo 法に利用される疑似乱数について書くことにします。

現在まで広く用いられてきた疑似乱数は大雑把に言うと (A) 線形合同法、(B)GFSR、(C) その他、に分類できます。特に (A)(B) が主流でした (過去形で書くのは、ここ数年でこれから紹介する MT が「無敵の疑似乱数」として普及してきているからです)。(A) は漸化式

$$y_{n+1} = a y_n + b \bmod M \quad a, b, M \in \mathbb{N} \tag{0.1}$$

によって 0 から $M-1$ までの自然数の列 $\{y_n\}_{n\in\mathbb{N}}$ を生成する方法で 1949 年に Lehmer によって開発されました [1]。y_n/M を 0 と 1 の間の乱数と見做します。UNIX 系の OS の標準ライブラリには `rand()`、`drand48()` 等が入っていて長い間標準的に使われてきましたがこれらは (A) に属します。(B) は漸化式

$$\mathbf{x}_{l+n} = \mathbf{x}_{l+m} + \mathbf{x}_l, \quad (l = 0, 1, 2, \dots) \tag{0.2}$$

によって $\mathbf{x}_n \in \mathrm{GF}(2)^w, (n = 0, 1, 2, \dots)$ を生成する方法です。$1 \le m \le n$ です。$\mathbf{GF}(2)^w$ 上の線形変換 T を一つ定め (これを調律変換という)$T\mathbf{x}_n$ を w 桁の 2 進小数と見做すことによって 0 と 1 の間の乱数を得ます。$t^n + t^m + 1 \in \mathrm{GF}(2)[t]$ が原始多項式の時、周期 $2^n - 1$ の疑似乱数列が得られるわけです。この方法は Tausworthe(1965)[9]、Lewis と Payne(1973)[2]、によって開発されました。この古典的な二つの方法以外にも非常に多くの方法が提案されてきましたがそれ等は (A)(B) に比べるとあまり使われてきませんでした。その理由が何かという問題ははっきりと答えの出る性質のものではありませんが、この歴史の示すことは、それ等の方法が既に普及している (A)(B) を使用者に捨てさせる程大きなアドバンテージを持つとは受け取られなかったということです。(A)(B) の美点で顕著なのはコンピュータ上で実現する時に非常に良い性質を有していることです。(A) は殆ど記憶領域を必要としません。また $M = 2^k$ とすれば比較的簡単な演算だけで周期 M の疑似乱数を実現できます。(B) は nw ビットの記憶領域とたった 3 箇所の記憶領域の参照、およびそれ等の間の排他的論理和演算だけで周期 $2^n - 1$ の疑似乱数を実現します。また専用回路で実現することも容易です。これらの性質は疑似乱数の高速生成を可能とします。このように (A)(B) は生成される疑似乱数の質とそれをコンピュータで発生させる速度、必要とする記憶領域の大きさが、他の方法に比べ、良いバランスを持っていたということです。これは疑似乱数の生成方法を評価する時に重要なポイントです。ある疑似乱数を変更して生成される数列の質が多少高くなったとしても生成速度の低下や必要とされる記憶領域の増大、そしてプログラムを変更する手間、がそれに見合うもので無ければそれはもとのものよりも良くなったとは言えないということです。この点については MT への高い評価と関係が深いので後にまた触れることにします。

MT の登場　このように線型合同法 (1949 年) と GFSR(1965 年) という非常に古い疑似乱数は改良されながら長い間使われつづけてきました。1990 年代になって幾つかの新らしい疑似乱数が登場しますが、そのなかの一つ、1992 年に松本-栗田 [3][4] により開発された Twisted GFSR は真に革新的であり重要なものでした。これは GFSR と同様に $\mathrm{GF}(2)^w$ の元の列 $\{\mathbf{x}_n\}_{n\in\mathbb{N}}$ を生成するものですが (0.2) の代わりに

$$\mathbf{x}_{l+n} = \mathbf{x}_{l+m} + \mathbf{x}_l A, \quad (l = 0, 1, 2, \dots) \tag{0.3}$$

を用います。ここで A は GF(2) の元を要素とする $w \times w$ 行列です。Twisted GFSR は nw ビットの記憶領域を使い $2^{nw} - 1$ という理論的に可能な最大の周期と n 次元均等分布性を可能とします。さらに行列 A として $\mathbf{x}A$ の計算が非常に高速になるようなものを取ることが出来るので生成速度も GFSR と殆ど同じです。同じ大きさの記憶領域を用いる GFSR の周期が $2^n - 1$ ですから比較にならない程長い周期を持っていることがわかります。つまり Twisted GFSR は非零項を沢山含む nw 次の特性多項式に対応するシフトレジスタを非常に高速に実現したものということが出来ます。

Twisted GFSR を改良したものが、松本-西村の Mersenne Twister (MT) です [5]。この論文には 3 つの重要なアイディアが述べられています。MT の漸化式、inversive-decimation 法 (巨大な特性多項式の原始性の判定法)、「欠けた配列」による実装、です。この 3 つは密接に関係して MT を成立させるものです。夫々について説明します。

MT の漸化式　MT では (0.3) の代わりに

$$\mathbf{x}_{k+n} = \mathbf{x}_{k+m} + \left(\mathbf{x}_k^u \middle| \mathbf{x}_{k+1}^l\right) A, \quad (k = 0, 1, 2, \dots) \tag{0.4}$$

という漸化式が使われます。$\left(\mathbf{x}_k^u \middle| \mathbf{x}_{k+1}^l\right)$ は $\mathbf{x}_k$ の上位 $(w-r)$ ビットと $\mathbf{x}_{k+1}$ の下位 r ビットを並べて得られる GF(2)^w の元です。($r \in \{0, \ldots, w-1\}$) こうして得られる数列の周期は最大 $2^{nw-r}-1$ になります。

inversive-decimation 法 この漸化式に対応する $nw-r$ 次の特性多項式が原始多項式であれば理論的な最大周期 $2^{nw-r}-1$ が達成されますが、次数が高く非零項を沢山ふくむ多項式の原始性の判定は一般には非常に困難です。($O(lp^2)$ の計算量を要する。$p = nw-r$, l は多項式の非零項の数で、MT の場合 10^4 以上。) この論文で $2^{nw-r}-1$ が素数 (すなわち Mersenne 素数) の時にこれを高速 ($O(p^2)$) に判定するアルゴリズムである inversive-decimation 法が開発されました。

欠けた配列 $\mathbf{x}_k^u, \mathbf{x}_{k+1}, \ldots \mathbf{x}_{k+n-1}$ を記憶領域上にこの順に並べ (0.4) の計算を二つのポインタの付け変えと単純なビット演算だけで可能にする方法です。これにより計算が非常に高速になりますが、もう一点重要なのは計算量が記憶領域の大きさ (すなわち、$nw-r$) に依らないということです。また、この実装は上述の inverse-decimation 法の高速な実現にも貢献しています。

これらの発明を駆使して [5] には $n = 624, w = 32, r = 31$ の場合の MT の具体例である MT19937 が提示されています。これは $2^{19937}-1$ という周期と 623 次元の均等分布を達成しています。高次元均等分布の計算には形式冪級数体の最短格子が使われました。MT19937 が達成した数字は他の最新の疑似乱数と比べても「冪が桁違い」であり、しかも生成速度は殆ど最速というとてつもないものでした。

MT の発明で重要なのは、MT19937 という実例を見付けたことよりもそれを見付ける方法を開発したことです。将来、コンピュータの性能が向上し MT19937 でも間に合わない位の巨大な計算が可能になった時には、[5] に従ってもっと大きな MT を見付ければ良いわけです。また、規模の小さな MT ならばこの論文の方法で非常に短時間に見付けることができます。[6] ではこれを利用して並列コンピュータの上で数千の独立した小規模の MT (小規模とはいっても他の疑似乱数よりは質が高い) を動的に発生させる—疑似乱数生成プログラムを動的に発生させる—という方法が提唱されています。並列コンピュータの上での疑似乱数発生は、現在、重要な問題ですが、これは今後、広く用いられるようになると思われます。

MT 以後 この様に既存の疑似乱数を遥かに上回る MT ですが、登場と同時に広まったわけではありませんでした。決して利用者が既存の疑似乱数に満足していたわけではないのですが、皮肉なことに疑似乱数研究の現状が疑似乱数の利用者の MT へのアクセスを妨げてしまったという側面があったように見えます。MT の登場から数年の間、疑似乱数と Monte Carlo 法の有名な会議に毎年参加していたときに私が見た情景は、(1) 古典的な或いは新らしいあまり性能の良くない疑似乱数をあるタイプの問題に適用したときに表われる問題点の報告 (もともと良くない疑似乱数なので良く問題が表われる)、(2) その様に指摘された問題を回避するための改良方法の報告、といった二つのタイプの報告が繰り返される様でした。MT は非常に性能が高いために問題が見付かりませんから (1) のタイプの報告はありません。(実際、松本-西村が [5] において発明時に報告している以外の新らしい話題は初期化の方法に関するものが一件あるだけです。) ということは (2) のタイプの研究もされませんから、MT の名前は疑似乱数の論文にはなかなか表われないこと

になります。勿論、沢山の疑似乱数を並べてそれらの性能を評価するという報告のなかではMTの圧倒的な成績が述べられていますがそのような報告は一回出るだけです。また、この分野の大家と呼ばれる人がMTの登場以後に書いた教科書の中で、何故か性能の悪い疑似乱数が新たに推奨されていたり、疑似乱数の専門家の中に、はっきりした理由も無く何故かMTを推奨しなかったりするひとがいたりしてその性能に比べてMTの普及は一時的に進みませんでした。疑似乱数の利用者は大抵、疑似乱数を専門としない人達ですからこのような障害があるとなかなかMTに辿り着くのは難しいわけです。

しかし、最近になってこのような状況は変りつつあります。性能の高い疑似乱数を必要とする利用者がなんとかMTを発見しそれを宣伝するという、主に利用者の側からの働きかけによりMTの知名度は非常に高くなり、様々なライブラリや標準においてMTが採用される様になりました。

高速であることの重要性　MTは長大な周期と高次元均等分布性をもつことだけで評価されているのではありません。これを極端に高速に発生させている、という点も非常に重要です。多くの科学技術計算では、計算時間全体の中に占める疑似乱数生成時間の割合が大きい場合が多いからです。これが重要であることは屡々見落とされているので注意をしたいと思います。

典型的な誤解は「コンピュータの速度と記憶領域の大きさは年々増大しているので疑似乱数生成プログラムの速度と実装効率の重要性は低くなっている」というものです。こういうことが、現実に権威ある疑似乱数の教科書に書かれているのですが、これは正しくありません。科学技術計算を実際にやっている人達の間ではよく理解されていることですが、コンピュータの性能が100倍になったときに何が起きるかというと、そのコンピュータを手にした人は100倍大きな問題に挑むことになるわけです。より大きな問題に対してはより沢山の疑似乱数が消費されるので疑似乱数の生成に費やされる時間は変りませんし、要求される疑似乱数の質の基準も高くなりますから疑似乱数の実装効率と生成速度は相変らず重要でありつづけることになります。

MTのこの極端な高速性は、MTを越えるものを作ることに対する高いハードルともなっています。例えば、MT19937を改良することを考えます。調律変換Tや行列Aをいじって発生させる数列の質を向上させることは可能でしょうが、おそらくは計算速度が低下します。この時、同じ計算時間で実現出来る、より周期の長いMTと比較してこの改良版MT19937がアドバンテージを持つかどうかは自明では無いように思われます。

非統計的検定　先程述べた様な疑似乱数研究の現場の有様をもたらしているものの一つに疑似乱数の質の理論的な評価基準が少ないという問題があります。その為、実際に疑似乱数を発生させて統計的検定をかける、という研究が非常に多く行なわれていますがそもそも決定論的に生成される数列に対してこのような検定は何か意味があるのか、等々の疑問もあります。松本-西村の最近の研究 [7] では、実際に数列を発生させずに、その疑似乱数をひっかけるような検定を作りだす方法が述べられています。その計算の鍵となるのはMac-Williams恒等式という符号理論における離散フーリエ反転公式です。これにより、生成される数列の空間という大きな空間上での分布の計算 (一般には計算困難) を、その双対空間 (数列が満たす関係式の空間、こちらは小さくなり計算可能) に帰着させることができます。

そしてこの中では、既存の様々な疑似乱数についてそれらの疑似乱数がどれ位の数の数列を発生させるとどれ位の確率で問題を生じるか、が計算されています。私はこの発表を聴いたときに松本さんに「コンピュータの性能向上曲線とこの結果を合わせると、この先、何年頃にそれぞれ

の疑似乱数について問題が生じたという論文が書かれるか、を予言することができるね」という冗談を言ったことを覚えています。

結び　大学入学以来の友人である松本眞さんの、この素晴らしいとしかいいようの無い疑似乱数研究の紹介をさせていただき大変に光栄に感じています。私の非力ではどれだけこの研究の価値が伝えれられたか疑問ですが。講演や論文に接したことのある方はご存知だと思いますが、松本さんの物事をやさしく説明する能力は大変なものです。拙稿によって松本さんの疑似乱数に興味をもたれた方には是非、参考文献にある松本さんの論文に当られることをお薦めします。

最後になりましたが、松本さん、受賞おめでとうございました。これからも (いろいろな意味で) 波乱を巻き起す研究を続けて下さることを信じています。

参考文献

[1] Lehmer, D. H. Mathematical methods in large-scale computing units. In *Proceedings of the Second Symposium on Large-Scale Digital Calculating Machinery* (1951), Harvard University Press, pp. 141–146.

[2] Lewis, T. G., and Payne, W. H. Generalized feedback shift register pseudorandom number algorithm. *Journal of the ACM 20* (1973), 456–468.

[3] Matsumoto, M., and Kurita, Y. Twisted GFSR Generators. *ACM Transactions on Modeling and Computer Simulations 2* (1992), 179–194.

[4] Matsumoto, M., and Kurita, Y. Twisted GFSR Generators II. *ACM Transactions on Modeling and Computer Simulations 4* (1994), 254–266.

[5] Matsumoto, M., and Nishimura, T. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Transactions on Modeling and Computer Simulations 8* (1998), 3–30.

[6] Matsumoto, M., and Nishimura, T. Dynamic Creation of Pseudorandom number generator. In *Monte Carlo and Quasi-Monte Carlo Methods 1998* (2000), H. Niederreiter and J. Spanier, Eds., Springer, pp. 56–69.

[7] Matsumoto, M., and Nishimura, T. A Nonempirical Test on the Weight of Pseudorandom Number Generators. In *Monte Carlo and Quasi-Monte Carlo Methods 2000* (2002), K. T. Fang, F. J. Hickernel, and H. Niederreiter, Eds., Springer, pp. 381–395.

[8] Metropolis, N. The beginning of the Monte Carlo method. *Los Alamos Science No. 15 Special Issue* (1987), 125–130.

[9] Tausworthe, R. C. Random numbers generated by linear recurrence modulo two. *Math. Comp. 19* (1965), 201–209.