IFT 3245
Simulation et modèles

Fabian Bastin
DIRO
Université de Montréal

Automne 2016

MRGs combinés

Considérons deux [ou plusieurs...] MRGs évoluant en parallèle:

$$\begin{aligned} x_{1,n} &= (a_{1,1}x_{1,n-1} + \cdots + a_{1,k}x_{1,n-k}) \mod m_1, \\ x_{2,n} &= (a_{2,1}x_{2,n-1} + \cdots + a_{2,k}x_{2,n-k}) \mod m_2. \end{aligned}$$

On définit les deux combinaisons:

$$\begin{aligned} z_n &:= (x_{1,n} - x_{2,n}) \mod m_1; \qquad u_n := z_n/m_1; \\ w_n &:= (x_{1,n}/m_1 - x_{2,n}/m_2) \mod 1. \end{aligned}$$

La suite $\{w_n,\ n \geq 0\}$ est la sortie d'un autre MRG, de module $m = m_1 m_2$, et $\{u_n,\ n \geq 0\}$ est presque la même suite si m_1 et m_2 sont proches. Peut atteindre la période $(m_1^k - 1)(m_2^k - 1)/2$.

Permet d'implanter efficacement un MRG ayant un grand m et plusieurs grands coefficients non nuls.

MRGs combinés

Pour accélérer la génération de point, il est possible de prendre tous les a_j non nuls égaux à a (Deng et Xu 2002). Alors, $x_n = a(x_{n-i_1} + \cdots + x_{n-k}) \mod m$. Une seule multiplication.

Les meilleurs générateurs ne jouissent cependant pas de cette propriété.
Tableaux de paramétres: L'Ecuyer (1999); L'Ecuyer et Touzin (2000).

MRG32K3a

$J = 2$, $k = 3$,
$m_1 = 2^{32} - 209$, $a_{11} = 0$, $a_{12} = 1403580$, $a_{13} = -810728$,
$m_2 = 2^{32} - 22853$, $a_{21} = 527612$, $a_{22} = 0$, $a_{23} = -1370589$.
Combination: $z_n = (x_{1,n} - x_{2,n}) \bmod m_1$.

Le générateur correspond à un MRG caractérisé par $k = 3$,
$m = m_1 m_2 = 18446645023178547541$, et les paramétres
$a_1 = 181696684712528925 57$, $a_2 = 3186860506199273833$,
$a_3 = 873861326439822262 2$. Sa période ρ vaut
$(m_1^3 - 1)(m_2^3 - 1)/2 \approx 2^{191}$.

MRG32K3a

```
#define norm 2.328306549295728e-10  /* 1/(m1+1) */
#define m1   4294967087.0
#define m2   4294944443.0
#define a12     1403580.0
#define a13n     810728.0
#define a21      527612.0
#define a23n    1370589.0

double  s10, s11, s12, s20, s21, s22;

double MRG32k3a ()
{
   long   k;
   double p1, p2;
```

```
    /* Component 1 */
    p1 = a12 * s11 - a13n * s10;
    k = p1 / m1;   p1 -= k * m1;   if (p1 < 0.0) p1 += m1;
    s10 = s11;   s11 = s12;   s12 = p1;

    /* Component 2 */
    p2 = a21 * s22 - a23n * s20;
    k  = p2 / m2;  p2 -= k * m2;   if (p2 < 0.0) p2 += m2;
    s20 = s21;   s21 = s22;   s22 = p2;

    /* Combination */
    if (p1 <= p2) return ((p1 - p2 + m1) * norm);
    else return ((p1 - p2) * norm);
}
```