Geradores de Números Aleatórios

Carlos Eduardo Costa Vieira
email:cadu@inf.puc-rio.br
Celso Carneiro Ribeiro
email:celso@inf.puc-rio.br
Reinaldo de Castro e Souza
Departamento de Engenharia Elétrica - PUC-Rio
email:reinaldo@ele.puc-rio.br

PUC-RioInf.MCC22/04 Junho, 2004

Abstract

This works presents a short introduction to random numbers generators which are based upon specific mathematical algorithms, sequential and deterministic. These algorithms are crucial for a whole range of applications. First, we discuss the mathematical definition of random number generators and the common techniques for producing the random sequence. Next, the desirable properties of good generators are discussed. Final remarks are made in the last Section.

Keywords: Random number generators, linear congruential generators, lagged Fibonacci generators, shift registers generators, hybrid generators.

Resumo

Este trabalho tem como objetivo fazer uma breve introdução sobre os geradores de números aleatórios que são algoritmos específicos, seqüenciais e determinísticos. Estes geradores são utilizados em uma grande faixa de aplicações. Primeiramente mostra-se a definição matemática e os tipos de geradores mais comumente utilizados na prática. Em seguida são apresentadas as propriedades desejáveis de um bom gerador. Considerações finais sobre os geradores são feitas na última Seção.

Palavras chaves: Geradores de números aleatórios, geradores congruentes lineares, geradores de atraso de Fibonacci, geradores de registradores de deslocamento, geradores híbridos.

Sumário

1 Introdução

O propósito dos geradores de números aleatórios é produzir uma sequência de números que aparentam ser gerados aleatoriamente de uma distribuição de probabilidade específica. Usualmente, um gerador de números aleatórios básico uniforme produz números que imitam variáveis aleatórias independentes da distribuição uniforme sobre o intervalo $[0, 1]$, (independentes e uniformemente distribuídas sobre o intervalo $[0, 1]$). As variáveis aleatórias de outras distribuições (normal, qui-quadrado, exponencial, poisson, etc.) são simuladas empregando-se transformações apropriadas aos números aleatórios uniformes [1].

Os geradores de números aleatórios são programas de computador cujo objetivo é imitar ou simular o comportamento típico de uma sequência de variáveis aleatórias independentes. São algoritmos específicos, sequencias e determinísticos (se inicializados em computadores ou momentos diferentes com o mesmo estado inicial ou semente, produzem a mesma sequência de números aleatórios). Os números gerados são, portanto, apenas pseudoaleatórios, mas com um leve abuso de linguagem são chamados de aleatórios. É importante então diferenciar [2]:

- Número aleatório - definido como exibindo aleatoriedade verdadeira, tais como o tempo entre *tics* de um contador *Geiger* exposto a um elemento radioativo.
- Número pseudoaleatório - definido como possuindo a aparência de aleatoriedade, contudo exibindo um padrão de repetição específico.
- Número quasi-aleatório - definido como preenchendo o espaço de solução sequencialmente. Por exemplo, considere o espaço inteiro $[0, 100]$. Uma sequência quasi-aleatória que preenche o espaço inteiro é $0, 1, 2, \ldots, 99, 100$. Outra sequência seria $100, 99, 98, \ldots, 2, 1, 0$. Uma terceira seria $23, 24, 25, \ldots, 99, 100, 0, 1, \ldots, 21, 22$. Sequências pseudoaleatórias que preenchem este espaço contêm os mesmos números, mas em ordem diferente.

Os geradores de números aleatórios são ingredientes cruciais para uma grande faixa de aplicações, tais como experimentos estatísticos, simulação de sistemas estocásticos, análises númericas com métodos de Monte Carlo, algoritmos probabilísticos, jogos de computador e criptografia, entre outros [1].

O objetivo deste trabalho é fazer uma breve introdução sobre os geradores de números aleatórios, organizada como se segue. A Seção 2 mostra a definição matemática dos geradores de números aleatórios. A Seção 3 apresenta os principais métodos utilizados na geração de números aleatórios. A Seção 4 apresenta as propriedades desejáveis em um bom gerador e, por último, a Seção 5 apresenta considerações finais sobre os geradores.

2 Definição Matemática

Matematicamente, um gerador de números aleatórios pode ser definido como uma estrutura (S, μ, f, U, g), onde S é um conjunto finito de estados, μ é uma distribuição de probabilidade em S usada para selecionar o estado inicial (semente) $s_0 \in S$, o mapeamento $f : S \to S$ é uma função de transição, U é um conjunto finito de símbolos de saída e $g : S \to U$ é uma função de saída [4].

O gerador opera da seguinte maneira: inicia-se do estado inicial s_0 (semente) e faz-se $u_0 = g(s_0)$. Procede-se então de acordo com a recorrência $s_n = f(s_{n-1})$, para $n \geq 1$. A saída no passo n é $u_n = g(s_n) \in U$. Assume-se que existam procedimentos eficientes para calcular f e g e para gerar a semente s_0 de acordo com μ. As variáveis u_n são os números aleatórios (ou observações) produzidos pelo gerador. Espera-se que as observações $u_1, u_2, \ldots, u_n$ comportem-se como variáveis aleatórias uniformemente distribuídas sobre o conjunto U. O conjunto U é freqüentemente um conjunto de inteiros da forma $\{0, 1, \ldots, m-1\}$ ou um conjunto finito de valores entre 0 e 1 para aproximar a distribuição $U[0, 1]$.

Como S é finito, o gerador eventualmente retornará a um estado já visitado (isto é, $s_{i+j} = s_i$ para algum $i, j \geq 0$). Então, $s_{n+j} = s_n$ e $u_{n+j} = u_n$ para todo $n \geq i$. O menor $j > 0$ para o qual isso acontece é chamado de período ρ. O período não pode exceder a cardinalidade de S. Um gerador de números aleatórios bem projetado nornalmente possui o perído ρ próximo de $|S|$, isto é, $\rho \approx 2^b$ se cada estado é representado por b bits.

3 Tipos de Geradores

Os tipos de geradores de números aleatórios mais comumente utilizados são [5, 6]:

- Geradores Congruentes Lineares,
- Geradores de Atraso de *Fibonacci*,
- Geradores de Registradores de Deslocamento, e
- Geradores Híbridos.

Para esses geradores, a teoria está bem desenvolvida e as implementações tem sido extensivamente testadas [7].

3.1 Geradores Congruentes Lineares

Os **Geradores Congruentes Lineares** são os algoritmos mais conhecidos e utilizados para a geração de números aleatórios. Possuem a seguinte fórmula de recorrência:

$$x_{n+1} = (ax_n + c)(mod\ m) \tag{1}$$

Esse tipo de gerador é também denominado de **Gerador Congruente Linear Misto**. O termo "misto" se refere ao fato desse gerador utilizar as operações de multiplicação e de adição por a e por c, respectivamente. O número aleatório x_{n+1} é gerado utilizando o número anterior x_n, as constantes inteiras a e c, a operação módulo m e uma semente inicial x_0 que deve ser fornecida por algum meio. Os princípios para a escolha dos números a, c, m e x_0 devem ser investigados cuidadosamente com a finalidade de obter o maior período possível. O fato de x_{n+1} depender somente de x_n minimiza as necessidades de armazenamento mas, ao mesmo tempo, impõe restrições no período. O correspondente número real pode ser determinado dividindo-se por m, obtendo-se números no intervalo $[0, 1)$ ou por $m - 1$, obtendo-se números no intervalo

$[0, 1]$. No caso onde $c = 0$, o gerador é denominado **Gerador Congruente Linear Multiplicativo**, cuja recorrência é

$$x_{n+1} = ax_n (mod\ m). \tag{2}$$

3.1.1 Gerador Congruente Linear Misto

Um Gerador Congruente Linear Misto bastante utilizado tem $m = 2^M$ e $c > 0$. O período completo $\rho = 2^M$ é obtido se, e somente se, $a \equiv 1\ mod\ 4$, isto é, se $a - 1$ é múltiplo de 4 e c é ímpar, sendo $c = 1$ freqüentemente escolhido [9]. Os bits de mais baixa ordem (menos significativos) possuem um padrão cíclico, decorrente da utilização de uma potência de 2 para m. Por exemplo, a sequência alterna entre números pares e ímpares. Apesar disto, o módulo com potência de 2 é freqüentemente utilizado, pois torna o processo de realização da operação módulo eficiente. Além do mais, os números reais obtidos da sequência inteira não possuem um padrão muito regular nos bits de mais alta ordem (mais significativos) e ainda são utilizados em muitas aplicações [2].

Exemplo - $x_{n+1} = (5x_n + 1)\ mod\ 2^4$ [2]

Sejam $a = 5$, $c = 1$, $m = 16$ e $x_0 = 1$. A sequência de inteiros aleatórios geradas por essa recorrência é: $1, 6, 15, 12, 13, 2, 11, 8, 9, 14, 7, 4, 5, 10, 3, 0, 1, 6, 15, 12, 13, 2, 11, 8, 9, 14, \ldots$

Observa-se que o período ρ é igual a 16 ($\rho = 2^M = 2^4 = 16$). Esta sequência possui o período mais longo possível e é uniforme, isto é, preenche completamente o espaço de inteiros entre 0 e 15. Exibe através do seu período o padrão de alternar inteiros ímpares e pares; assim, o dígito binário mais à direita exibe o padrão regular $1, 0, 1, 0, \ldots$ Também, ao selecionar qualquer semente inicial entre 0 e 15, desloca-se ciclicamente a sequência acima.

3.1.2 Gerador Congruente Linear Multiplicativo

Pelo fato de não realizarem a operação de adição, os Geradores Multiplicativos são mais eficientes do que os Geradores Mistos em termos de tempo de processamento necessários para realizarem uma determinada tarefa. Também, eficiência adicional pode ser obtida escolhendo-se $m = 2^M$, tornando assim a operação módulo mais simples. Desta maneira, existem dois tipos de Geradores Multiplicativos, aqueles no qual $m = 2^M$ e aqueles no qual $m \neq 2^M$ [6].

O principal argumento em favor da escolha de $m = 2^M$ é a facilidade de calcular a operação módulo, porém, tais geradores não possuem um período completo. O seu período máximo é $\rho = 2^{M-2}$, isto é, um quarto do valor de m, e é obtido, se, e somente se, $a \equiv 3\ mod\ 8$ ou $a \equiv 5\ mod\ 8$ e a semente inicial é ímpar [9]. Os valores de a congruentes a 5 módulo 8 são preferíveis aos valores de a congruentes a 3 módulo 8, pois testes de independência com esses multiplicadores mostram uma maior uniformidade do primeiro em relação ao segundo no espaço bi-dimensional [2, 8]. Os bits de mais baixa ordem dos números aleatórios obtidos por esses geradores também possuem um padrão cíclico [6].

Exemplo - $x_{n+1} = 5x_n \ mod \ 2^5$ [6]

Sejam $a = 5$, $c = 0$, $m = 2^5 = 32$ e $x_0 = 1$. A sequência de inteiros aleatórios geradas por essa recorrência é: $5, 25, 29, 17, 21, 9, 13, 1, 5, 25, \ldots$ O período ρ é 8, um quarto do valor máximo possível $m = 32$. Seja agora a semente par $x_0 = 2$. A sequência gerada é: $10, 18, 26, 2, 10, \ldots$, com o período igual a 4. Considere agora o gerador $x_{n+1} = 7x_n \ mod \ 2^5$, com $a = 7$, $c = 0$, $m = 32$ e $x_0 = 1$. A sequência gerada é: $7, 17, 23, 1, 7, 17, \ldots$ Novamente, o período é somente 4. Portanto, ambas as condições do multiplicador a e da semente ímpar são necessárias para atingir o período máximo.

Uma solução para o problema de se ter um pequeno período é utilizar o módulo $m \neq 2^k$, como, por exemplo, fazer $m = p$, sendo p primo. O período máximo deste gerador é $p-1$ e é obtido se, e somente se, a é uma raiz primitiva do módulo p e x_0 é diferente de 0 [9]. Por definição, a é uma raiz primitiva de p se, e somente se, $a^n \ mod \ p \neq 1$ para $n = 1, 2, \ldots, p-2$ [6].

Exemplo - $x_{n+1} = 3x_n \ mod \ 31$ [2]

Sejam $a = 3$, $c = 0$, $m = p = 31$ e $x_0 = 1$. A sequência de inteiros aleatórios geradas por essa recorrência é: $1, 3, 9, 27, 19, 26, 16, 17, 20, 29, 25, 13, 8, 24, 10, 30, 28, 22, 4, 12, 5, 15, 14, 11, 2, 6, 18, 23, 7, 21, 1, \ldots$ O período desse gerador é 30, o máximo possível, pois o multiplicador $a = 3$ é uma raiz primitiva de 31, desde que o menor valor positivo para n para o qual $3^n \ mod \ 31 = 1$ é igual a 30. Ao se utilizar $a = 5$, obtém-se a seguinte sequência: $1, 5, 25, 1, 5, 25, \ldots$, onde o período é somente 3. O multiplicador $a = 5$ não é uma raiz primitiva de 31, já que $5^3 \ mod \ 31 = 125 \ mod \ 31 = 1$.

3.1.3 Estrutura Lattice

Um fenômeno interessante que ocorre com os geradores lineares são as estruturas *lattice* das t-tuplas obtidas de números aleatórios consecutivos. Considere a sequência $\{u_n : 0 \leq n < P\}$ no intervalo $[0, 1)$, onde P é o período. Ao gerar o gráfico, por exemplo, do conjunto de todos os pares adjacentes $\{(u_n, u_{n+1}); 0 \leq n < P\}$, espera-se cobrir o espaço bi-dimensional com uma boa uniformidade. Pode-se suspeitar de um gerador que produza pontos no plano agrupados na metade para baixo ou possuam alguma ordem clara da esquerda para a direita. Uma característica dos geradores lineares é o fato de que os pontos selecionados desta maneira formam linhas com aparência regular facilmente percebidas quando pontos suficientes são vistos em uma escala apropriada. Estas observações podem ser estendidas para o espaço tridimensional, onde o conjunto de todas as triplas adjacentes $\{(u_n, u_{n+1}, u_{n+2}); 0 \leq n < P\}$ estão localizadas em planos diferentes. Generalizando, a estrutura *lattice* significa que todos os pontos adjacentes no espaço t-dimensional $\{(u_n, \ldots, u_{n+t-1}); 0 \leq n < P\}$ estão localizados em uma família de hiperplanos paralelos equidistantes [10].

As Figuras 1, 2 e 3 mostram um exemplo no plano para ilustrar esse comportamento [2]. Todos os pares de números consecutivos no período completo do gerador $x_{n+1} = ax_n \ mod \ 509$ foram plotados. Para a Figura 1, o multiplicador a é 10, para a Figura 2, o multiplicador a é 128 e para a Figura 3, o multiplicador a é 108. Assim, para $m = 509$, os pontos mostram o efeito

da escolha do multiplicador a sobre a estrutura *lattice* do gerador. Na Figura 1, pode-se notar que os pontos formam um conjunto de linhas. A melhor situação é ter o máximo espaçamento entre as linhas tão pequena quanto possível. Na Figura 2, nota-se que o espaçamento entre as linhas é maior, então claramente o quadrado unitário não é bem coberto pelo conjunto de pontos. A Figura 3 mostra o efeito de uma boa escolha do multiplicador a, para o qual o máximo espaçamento entre as linhas é claramente menor do que nos dois primeiros casos e onde os pontos cobrem o quadrado unitário com uma uniformidade quase ótima.

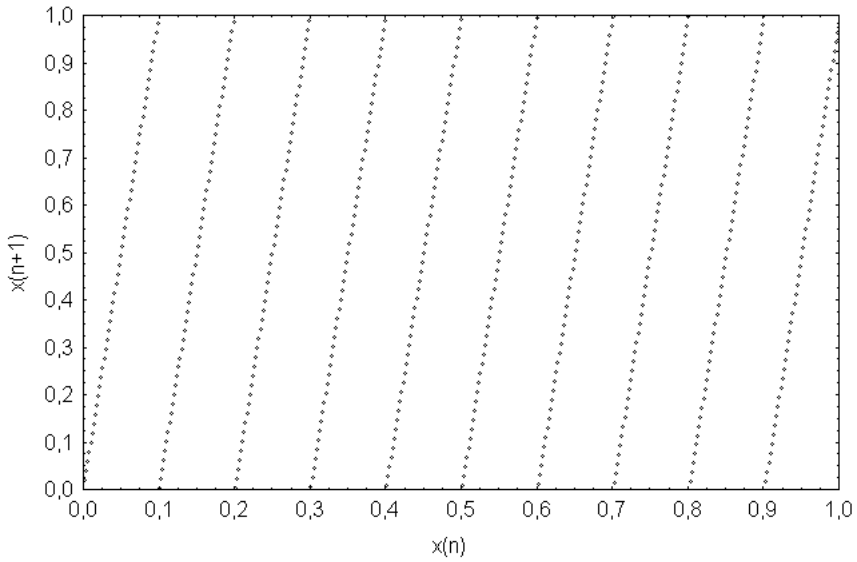

Figura 1: Pares $\{(u_n, u_{n+1}); 0 \leq n < 509\}$ para $x_{n+1} = 10x_n \ mod \ 509$

3.1.4 Exemplos de Geradores Congruentes Lineares Conhecidos

Alguns geradores bem conhecidos pertencem a essa classe. Denotando-se cada gerador por $GCL(a, c, m, x_0)$, pode-se destacar [11]: o gerador *RANDU GCL* $(65539, 0, 2^{31})$, o gerador do *SIMSCRIPT* $GCL(630360016, 0, 2^{31}-1)$, o gerador do *NAG* $GCL(13^{13}, 0, 2^{59}, 123456789(2^{32}+1))$, o gerador do *Maple GCL* $(427419669081, 0, 10^{12}-11, 1)$, o gerador do sistema *Ansi-C* $GCL(1103515$ $245, 12345, 2^{31}, 12345)$ e o gerador *padrão mínimo* $GCL(16807, 0, 2^{31}-1)$. Exemplo desse último é o gerador congruente multiplicativo de Linus Schrage [3].

3.2 Geradores de Atraso de Fibonacci

Essa classe de geradores possui esse nome por causa da sua similaridade com a familiar sequência de Fibonacci, definida por $x_n = x_{n-1} + x_{n-2}$, com $x_0 = x_1 = 1$.

Generalizando essa fórmula tem-se a família de geradores da forma:

$$x_n = x_{n-l} \ op \ x_{n-k} \tag{3}$$

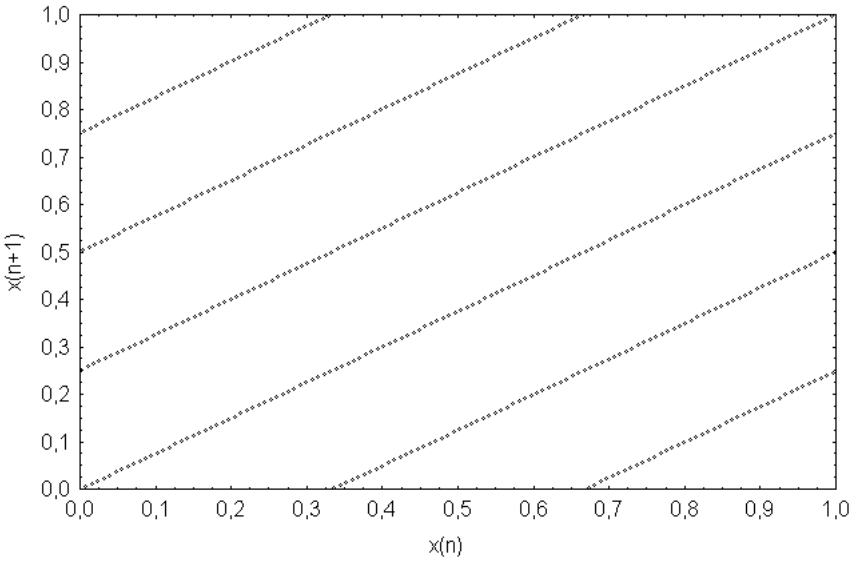

Figura 2: Pares $\{(u_n, u_{n+1}); 0 \leq n < 509\}$ para $x_{n+1} = 128x_n \ mod \ 509$

onde $l > k > 0$ e l valores iniciais $x_0, \ldots x_{l-1}$ são necessários para calcular o próximo elemento da sequência. Nesta expressão, os atrasos (*lags*) são k e l, e o valor corrente de x é determinado pelos valores nas posições x_k e x_l da sequência. As possíveis operações *op* são: adição, subtração e multiplicação módulo m e a operação "ou exclusivo". Para a maioria das aplicações de interesse, m é uma potência de 2, isto é, $m = 2^M$ [2].

Para as operações de adição e subtração módulo m, por exemplo, através de uma escolha apropriada de k e l, o período ρ deste gerador é igual a $(2^l - 1) \times 2^{M-1}$. A única condição para os primeiros l valores de x é que pelo menos um deles deve ser ímpar. O valor de m não limita o período do gerador, como no caso dos geradores congruentes lineares. Portanto, longos períodos são possíveis. Como exemplo, seja $l = 607$, $k = 273$, $m = 2^{31}$ e a operação subtração módulo m. O período desse gerador é $(2^{607} - 1) \times 2^{31-1} \approx 2^{101}$, se pelo menos um membro da sequência inicial $x_0, \ldots, x_{606}$ for ímpar. A principal desvantagem é o fato de que l palavras de memória devem ser armazenadas para o cálculo do próximo número aleatório, enquanto um gerador linear necessita de somente uma, correspondente ao último valor de x gerado [2].

3.3 Geradores de Registradores de Deslocamento

Os **Geradores de Registradores de Deslocamento** ou **Geradores *Tausworthe*** consideram os bits de uma palavra de computador como os elementos de um vetor binário, na qual, iterativamente, através de transformações lineares geram sequências de vetores binários interpretadas como sequências de números inteiros aleatórios uniformes. Seja $\beta_{(1\times n)} = (b_1, b_2, \ldots, b_n)$ um vetor binário e $T_{(n\times n)}$ uma matriz binária. Para produzir iterativamente os vetores binários, esses geradores utilizam a sequência $\beta, \beta T, \beta T^2, \beta T^3, \ldots$, sendo toda a aritmética realizada sobre a operação módulo 2. A matriz binária T é escolhida

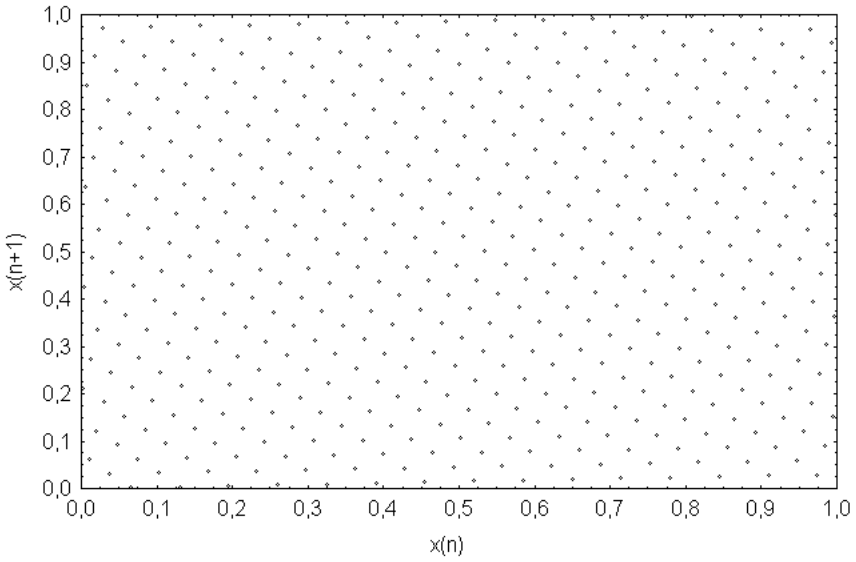

Figura 3: Pares $\{(u_n, u_{n+1}); 0 \leq n < 509\}$ para $x_{n+1} = 108x_n \ mod \ 509$

de tal maneira que o produto βT seja realizado através de operações elementares do computador. Uma boa escolha [5, 7], é $T = (I + R^s)(I + L^t)$, onde n, $s < n$ e $t < n$ são inteiros, $I_{(n \times n)}$ é a matriz identidade e R e L são dados por:

$$R = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 \\ & \vdots & \vdots & & & \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}, \ L = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ & \vdots & \vdots & & & \\ 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 \end{pmatrix}$$

A multiplicação βR desloca o vetor β um bit para a direita, isto é, transforma o vetor $\beta = (b_1, b_2, \dots, b_{n-1}, b_n)$ em $\beta = (0, b_1, b_2, \dots, b_{n-1})$ e a multiplicação βL desloca o vetor β um bit para a esquerda, isto é, transforma o vetor $\beta = (b_1, b_2, b_3, \dots, b_n)$ em $\beta = (b_2, b_3, \dots, b_n, 0)$. Assim, o produto $\beta T = \beta(I + R^s)(I + L^t)$ pode ser produzido realizando $\beta \leftarrow \beta \oplus \beta R^s$ com s deslocamentos à direita e uma operação "ou exclusivo" bit a bit, seguidos de $\beta \leftarrow \beta \oplus \beta L^t$ com t deslocamentos à esquerda e uma operação "ou exclusivo" bit a bit. O máximo período possível de um gerador desse tipo é $2^n - 1$, que é o número de vetores binários não-nulos $(1 \times n)$. Para tais geradores, qualquer vetor inicial não nulo pode servir como semente [5].

Exemplo

Sejam $\beta = (1, 1, 0, 1, 0)$, $s = 2$, $t = 3$ e $n = 5$. Primeiramente, desloca-se $s = 2$ bits à direita do vetor β. Então, $X_1 = (0, 0, 1, 1, 0)$. Faz-se a seguir uma operação "ou exclusivo" bit a bit entre X_1 e β. Assim $X_2 = X_1 \oplus \beta = (1, 1, 1, 0, 0)$. Desloca-se, agora, o vetor X_1 $t = 3$ bits para a esquerda. Então, $X_3 = (0, 0, 0, 0, 0)$. Por último, faz-se uma operação "ou exclusivo" bit a bit en-

tre X_2 e X_3. O próximo número aleatório gerado é $X_4 = X_2 \oplus X_3 = (1, 1, 1, 0, 0)$. A partir de X_4 e fazendo o procedimento acima três vezes seguidas tem-se os vetores binários: (0,0,0,1,1), (1,1,0,1,1), (1,0,1,0,1).

3.4 Geradores Híbridos

É possível combinar dois ou mais geradores com a finalidade de produzir um gerador resultante melhor. Além de fortes evidências empíricas, os geradores híbridos possuem algum suporte teórico. Primeiro, na maioria dos casos, o período do gerador híbrido é muito maior do que de cada um dos geradores componentes. Segundo, existem resultados teóricos sugerindo que algumas formas de geradores híbridos geralmente possuem um melhor comportamento estatístico [10]. Por exemplo, sejam x_n e y_n duas sequências de números aleatórios definidas no conjunto finito 0 e $m-1$. Pode-se combiná-las para produzir a sequência $z_n = (x_n \ op \ y_n) \ mod \ m$, onde *op* denota algum operador binário $(+, -, *, \oplus)$ com a sequência z_n sendo também definida no conjunto 0 e $m-1$. Sob algumas condições, pode-se mostrar que a sequência resultante é mais uniforme e independente do que as sequências componentes [5]. Contudo, combinar geradores sem um suporte teórico pode produzir maus geradores [11]. Na versão mais simples dessa técnica, combinam-se dois geradores adicionando-se as sequências x_n e y_n. Assim: $z_n = (x_n + y_n) \ mod \ 1, n \geq 0$. Se os dois geradores são escolhidos apropriadamente, o período do gerador híbrido será o produto do período dos geradores componentes [11].

Uma outra técnica usada para combinar geradores é denominada **Shuffling**. Utiliza-se uma sequência como índice para decidir quais dos números gerados por outra sequência deveriam ser escolhidos. Muitos procedimentos de *Shuffling* têm sido definidos na literatura e a técnica mais popular é conhecida como algoritmo *M*. Essa técnica trabalha da seguinte forma: seja, por exemplo, um vetor de tamanho 100 preenchido com os números aleatórios de uma sequência x_n. Para gerar um número aleatório, gera-se um novo y_n (entre 0 e $m-1$) e obtém-se um índice $i = 1 + 100y_n/m$. O valor no i-ésimo elemento do vetor será retornado como o próximo número aleatório. Por último, um novo valor de x_n é gerado e armazenado na i-ésima localização [6]. A técnica de *Shuffling* não é bem entendida teoricamente e possui algumas desvantagens práticas. Por exemplo, não existe uma maneira óbvia eficiente de gerar um número aleatório sem ter que passar por todos os estados intermediários [10].

4 Propriedades Desejáveis de um Bom Gerador

Um bom gerador de números aleatórios deveria possuir as seguintes propriedades [5, 8, 9, 10, 12]:

- Uniformidade - A sequência de números aleatórios deve passar em testes estatísticos com a finalidade de verificar a uniformidade da distribuição.
- Indepedência - Subsequências da sequência completa $u_0, u_1, \ldots$ devem ser independentes. Por exemplo, membros da subsequência par $u_0, u_2, u_4, \ldots$ devem ser independentes dos seus vizinhos ímpares $u_1, u_3, \ldots$ Desta maneira, a sequência de pares (u_{2n}, u_{2n+1}) deve ser uniformemente distribuída no quadrado unitário. Em geral, números aleatórios são freqüen-

temente utilizados em amostras no espaço $t-$dimensional, com $t = 2, 3, 4, \ldots$. Então a sequência de $t-$tuplas $(u_{tn}, u_{tn+1}, \ldots, u_{tn+t-1})$ deve ser uniformemente distribuída no cubo $t-$dimensional $[0, 1]^t$, pelo menos para pequenos valores de t.

- Período longo - O gerador deve possuir um período longo. Idealmente, o gerador não deve repetir valores. Na prática, a repetição deve ocorrer somente depois da geração de um grande conjunto de números aleatórios.

- Facilidade de implementação e eficiência - Os geradores devem ser fáceis de serem implementados em uma linguagem de alto nível e eficientes, isto é, utilizar poucas operações aritméticas para gerar cada número aleatório, usar todas as capacidades vetoriais/paralelas disponíveis na máquina e minimizar *overheads* tais como chamadas a subrotinas;

- Repetição - Os geradores devem possuir a habilidade de repetir exatamente a mesma sequência de números aleatórios, pois isto é importante em procedimentos de teste e desenvolvimento de programas;

- Portabilidade - Os geradores deverm ser portáveis, isto é, gerar exatamente a mesma sequência de números aleatórios em duas máquinas diferentes, possivelmente com diferentes tamanhos de palavras.

- Subsequências disjuntas - Os geradores devem dispor de métodos eficientes para gerar um número aleatório sem ter que passar por todos os estados intermediários. Essa característica é importante para a utilização dos geradores de números aleatórios em ambientes de processamento paralelo, permitindo, assim, particionar a sequência em subfluxos disjuntos, criando um número arbitrário de geradores virtuais (um por subfluxo) a partir de um único fluxo. Como exemplo, em um Gerador Congruente Linear Multiplicativo, pode-se utilizar a relação $x_{n+\nu} = (a^\nu \ mod \ m)x_n \ mod \ m$ para "saltar" ν valores na sequência [13]. Deseja-se calcular x_8 a partir de $x_{n+1} = 5x_n \ mod \ 37$ sem ter que gerar todos os estados intermediários $x_1, x_2, \ldots, x_7$. Sejam $a = 5$, $c = 0$, $m = 37$ e $x_0 = 1$. A sequência de inteiros aleatórios geradas por essa recorrência é: $1, 5, 25, 14, 33, 17, 11, 18, 16, 6, 30, \ldots$. Assim, o objetivo é gerar o número aleatório $x_8 = 16$. Com isso $x_{n+8} = (a^8 \ mod \ 37)x_n \ mod \ 37 = (390625 \ mod \ 37)x_n \ mod \ 37 = 16x_n \ mod \ 37$. Portanto, $x_8 = 16x_0 \ mod \ 37 = 16$.

5 Considerações Finais

Este trabalho abordou em linhas gerais os geradores de números aleatórios, mostrando sua definição, as principais técnicas utilizadas na geração de números aleatórios e as propriedades desejáveis de um bom gerador.

Criar um bom gerador de números aleatórios não é uma tarefa trivial. Deve-se salientar que é difícil ou quase impossível obter um gerador que satisfaça todos os requisitos mostrados na seção 4. Geralmente, um gerador com boas propriedades teóricas, um período longo e que passe em diversos testes estatísticos é satisfatório para a maioria das aplicações.

Referências

[1] **P. L'Ecuyer**, "Random Numbers", *Int. Encyc. Social and Behavioral Sciences*, p. 1-9, 4 June, 2001.

[2] **Random Numbers**, "Computational Science Education Project", 1995, On-line document at *http* : *//csep*1.*phy.ornl.gov*/*CSEP*/*PS_FILES*/*RN.PS*, last visited on March 10, 2004.

[3] **L. Schrage**, "A More Portable Fortran Random Number Generator", *ACM Transactions on Mathematical Software*, v. 5, p. 132-138, 1979.

[4] **P. L'Ecuyer**, "Uniform Random Number Generation", *Annals of Operations Research*, v. 53, p. 77-120, 1994.

[5] **G. Marsaglia**, "A Current View of Random Number Generators", *in: Billard, Ed., Computer Science and Statistics: Proc. 16th Symposium on the Interface (North-Holland, Amsterdam)*, p. 3-10, 1985.

[6] **R. Jain**, "The Art of Computer Systems Performance Analysis - Techniques for Experimental Design, Measurement, Simulation, and Modeling", Wiley, 1991.

[7] **W. F. Eddy**, "Random Number Generators for Parallel Processors", *Journal of Computational and Applied Mathematics*, v. 31, p. 63-71, 1990.

[8] **S. L. Anderson**, "Random Numbers Generators on Vector Supercomputers and Other Advanced Architectures", *SIAM Review*, v. 32, p. 221-251, 1990.

[9] **D. E. Knuth**, "The Art of Computer Programming, v. 2 - Seminumerical Algorithms", Third Edition, Addison Wesley Longman, 1998.

[10] **P. L'Ecuyer**, "Random Numbers for Simulation", *Communications of the ACM*, v. 33, p. 85-97, 1990.

[11] **P. Hellekalek**, "Good Random Number Generators Are (Not So) Easy to Find", *Mathematics and Computers in Simulation*, v. 46, p. 485-505, 1998.

[12] **R. P. Brent**, "Uniform Random Number Generators for Supercomputers", *Proc. Fifth Australian Supercomputer Conference*, pp. 95-104, Melbourne, December 1992, On-line document at http://www.comlab.ox.ac.uk/oucl/work/richard.brent/pub/pub132.html/rpb132.pdf, last visited on June 23, 2004.

[13] **P. L'Ecuyer**, "Random Number Generation", *Handbook of Simulation (Chapter 4)*, Ed.: Jerry Banks, Wiley, 1998.