# GERANDO ORIENTAÇÕES ACÍCLICAS COM ALGORITMOS PROBABILÍSTICOS DISTRIBUÍDOS

**Gladstone M. Arantes Jr,**
Programa de Engenharia de Sistemas e Computação-COPPE,
Universidade Federal do Rio de Janeiro
Caixa Postal 68511, 21941-972, Rio de Janeiro, RJ, Brasil
glads@cos.ufrj.br

**Felipe M. G. França,**
Programa de Engenharia de Sistemas e Computação-COPPE,
Universidade Federal do Rio de Janeiro
Caixa Postal 68511, 21941-972, Rio de Janeiro, RJ, Brasil
felipe@cos.ufrj.br

**Carlos A. Martinhon**
Instituto de Computação, Universidade Federal Fluminense
Rua Passo da Pátria 156, Bloco E, Sala 303,
São Domingos, 24210-230, Niterói, RJ, Brasil
mart@dcc.ic.uff.br

**Resumo—**
Este artigo discute dois novos algoritmos distribuídos probabilísticos para a geração de orientações acíclicas em um sistema distribuído anônimo de topologia arbitrária. Os algoritmos são analisados tanto em termos de correção e complexidade, quanto velocidade de convergência. Em particular, é demonstrado que o novo algoritmo chamado *Alg-Arestas*, é capaz de produzir orientações acíclicas quase instantaneamente, isto é, em menos de dois passos com alta probabilidade. Duas aplicações para essa forma de quebra de simetria serão discutidas: inicialização do *Escalonamento por Reversão de Arestas (ERA)*, um simples e poderoso algoritmo de escalonamento distribuído, e uma estratégia de distribuição de *uploads* em redes de computadores

**Palavras-chave—** algoritmos distribuídos probabilísticos, quebra de simetria, sistemas anônimos.

**Abstract—**
This paper discusses two new randomized distributed algorithms for the generation of acyclic orientations upon anonymous distributed systems of arbitrary topology. Both algorithms are analysed in terms of correctnes and complexity as its rate of convergency. In particular, it is shown that the new algorithm called *Alg-Arestas* is able to produce acyclic orientations *quasi* instantaneously, i.e., in less than two steps with high probability. Two applications of this form of symmetry breaking will be discussed: initialization of *Scheduling by Edge Reversal* (SER), a simple and powerful distributed scheduling algorithm, and a strategy for distributed uploading in computer networks.

**Keywords—** randomized distributed algorithms, symmetry breaking, anonymous systems.

## I. INTRODUÇÃO

Neste trabalho apresentaremos dois novos algoritmos probabilísticos que geram orientações acíclicas em sistemas distribuídos anônimos, i.e., sistemas onde não existam identificações globais dos seus componentes. Sendo um dos possíveis métodos de quebra de simetria (criação de conjuntos independentes maximais e coloração são outras formas, todas equivalentes entre si em algum nível), esta é uma aplicação muito útil, com muitos trabalhos correlatos publicados, incluindo algoritmos paralelos, distribuídos e seqüênciais, tanto determinísticos quanto probabilísticos [GOL 87] [GOL 88] [ITA 90] [LUB 86] [PAN 92] [SZE 93] [FRA 91]. As maiores motivações para a concepção de algoritmos probabilísticos, em oposição aos algoritmos puramente determinísticos, estão na simplicidade de suas implementações e na rápida geração de soluções com características distintas. Esta diversidade torna-se particularmente interessante em aplicações baseadas em orientações acíclicas, como discutiremos adiante.

Estudamos correção, complexidade e velocidade de convergência dos dois novos algoritmos probabilísticos distribuídos, *Alg-Viz* e *Alg-Arestas*. Lidamos com algoritmos probabilísticos onde, a cada nova execução, obtém-se uma saída distinta com tempo de processamento também distinto. Logo, uma vez garantida que as orientações geradas sejam de fato acíclicas (correção), o tempo de processamento do algoritmo pode deve ser expresso como uma *variável aleatória* com *valor esperado* indicando sua complexidade esperada. Ambos os algoritmos fazem o uso de *dados*, viciados ou não, com $f \geq 2$ faces. Para $f = n$, foi determinado que o algoritmo *Alg-Viz* é linear no número de processadores; mais especificamente, possui custo de execução $ne$, onde $n$ representa o número de processadores e $e \approx 2{,}17$. A análise do novo algoritmo chamado *Alg-Arestas* foi baseada na noção de *recorrência probabilística* [KAR 94] e resultou em complexidade esperada $O(\lceil \log_f m \rceil)$ para um grafo conexo qualquer com $|E|=m$ arestas e $f \geq 2$.

Na Seção II, introduzimos conceitos básicos: sistemas distribuídos anônimos e sua representação em termos de grafos. Na Seção III, apresentamos dois algoritmos básicos de *Calabrese/França* [CAL 94] [CAL 97] e introduzimos os dois algoritmos de geração de orientações acíclicas (*Alg-Viz* e *Alg-Arestas*), discutindo sua convergência, complexidade esperada e velocidade de convergência. Algumas aplicações envolvendo a determinação de orientações acíclicas no contexto de sistemas anônimos são analisadas na Seção IV. Finalmente, nossas conclusões, incluindo futuros desenvolvimentos, são expostas na Seção V.

## II. CONCEITOS BÁSICOS

### A. Sistemas Anônimos

Um sistema distribuído pode ser descrito, sucintamente, como um conjunto de nós de processamento organizados de tal forma que cada nó está conectado a um subconjunto de outros nós por canais de comunicação. A comunicação entre cada par de nós se dá através de envio de mensagens. Comumente, assume-se que as conexões são bidirecionais, o que significa que um nó conectado a outro tanto pode enviar mensagens a ele quanto receber. Esta é uma definição bastante genérica, podendo ser aplicada para uma enorme gama de sistemas.

Um sistema distribuído é considerado *anônimo* quando não existe um identificador global que diferencie um nó de outro. No nosso caso, estaremos supondo que não exista nenhuma informação global acerca do sistema, ou seja, que não há disponível, *a priori*, informações como o número total de nós, o número de canais ou ainda qual a topologia do sistema. Esse tipo de sistema apresenta grandes dificuldades para ser manipulado [BAR 89] [BAR 96] [ARA 99] [ARA 01].

## B. Notação Adotada

Todo sistema distribuído pode ser modelado como um grafo $G=(V, E)$. Nesse grafo, $V$, de cardinalidade $n$, é o conjunto dos nós do sistema e $E$, de cardinalidade $m$, é o conjunto de arestas do tipo $(n_i,n_j)$, onde $n_i$ e $n_j$ são nós de $V$. Se $(n_i,n_j) \in E$, então existe um canal de comunicação bidirecional entre $n_i$ e $n_j$.

### III. OS ALGORITMOS DISTRIBUÍDOS

Descrevemos e analisamos aqui os dois algoritmos distribuídos probabilísticos de nosso interesse (*Alg-Viz* e *Alg-Arestas*). Estes dois algoritmos foram desenvolvidos a partir das idéias básicas contidas em [CAL 94] e [CAL 97], traduzidas pelos algoritmos de *Calabrese/França*, respectivamente nas versões *não-polarizada* e *polarizada.* Eles são apresentados logo a seguir e precedem a apresentação dos algoritmos *Alg-Viz* e *Alg-Arestas*.

## A. Calabrese/França

No algoritmo de Calabrese/França não-polarizado utiliza-se um gerador de números aleatórios que pode gerar 0 ou 1 com probabilidade 1/2. Por isso, dizemos tratar-se de uma *moeda equilibrada* ou *não-polarizada*.

O algoritmo executa sincronamente. Um nó é dito *probabilístico* se ele ainda possui arestas incidentes não orientadas e continua tomando parte nos sorteios, ou *determinístico*, no caso de não participar mais por já ter tido todas as arestas incidentes orientadas. Em cada passo do algoritmo todos os nós probabilísticos lançam uma moeda, obtendo 0 ou 1 (aqui representadas por $moeda_i$ para $i$=1,2,..,$n$). Um nó que obtiver 1 e cujos vizinhos probabilísticos restantes tiverem obtido 0 irá orientar todas as suas arestas ainda não orientadas na sua direção. O algoritmo continua até que todas as arestas sejam orientadas.

O algoritmo de Calabrese/França não-polarizado é bastante ineficiente se a probabilidade de se obter 0 ou 1 no sorteio for igual a 1/2. Por isso, em sua versão polarizada (que também opera sincronamente) as moedas serão *viciadas*, da seguinte forma: $\Pr\{moeda_i=1\}= 1/(|N(n_i)|+1)$ e $\Pr\{moeda_i=0\}=1-\Pr\{moeda_i=1\}$, onde $|N(n_i)|$ representa a cardinalidade do conjunto de vizinhos probabilísticos $N(n_i)$ do nó $n_i$. Desta forma, intuitivamente, em uma dada vizinhança, apenas um dos nós tende a obter 1 no sorteio.

### A.1 Análise de Complexidade

O tempo esperado de processamento é a medida do número de passos que o algoritmo precisa dar para que este convirja (termine), ou seja, até que todos os nós se tornem determinísticos. É fácil ver que o algoritmo de *Calabrese/França* [CAL 94] para moedas equilibradas é bastante ineficiente. Sucintamente, pode-se mostrar neste caso que, para grafos completos, aproximadamente $O(2^n)$ passos serão necessários para orientação completa de todas as arestas. A análise do tempo de processamento utilizando grafos completos é interessante já que, além de mais simples, um grafo completo representa, na verdade, a pior instância para o problema. Se $S_i$ é um evento indicando que $n_i \in$ V obteve 1 e os demais nós probabilísticos obtiveram 0 então, no máximo, um evento $S_i$ (para algum $i\in\{1,2..,n\}$). ocorrerá de cada vez. Para mostrar esse fato, seja S um evento indicando a união de $n$ eventos disjuntos $S_i$ (para $i$=1,..,$n$).

Tem-se então que:

$$\Pr(S) = \Pr\left(\bigcup_{i=1}^{n} S_i\right) = \sum_{i=1}^{n} \Pr(S_i) = \sum_{i=1}^{n} \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)^{n-1} = \frac{n}{2^n}$$

Observe que o número de repetições pode ser modelado por uma variável aleatória com distribuição geométrica e valor esperado $2^n.n^{-1}$. Assim, $O(2^n.n^{-1})$ repetições serão necessárias (valor esperado) até que algum dos eventos $S_i$ ocorra. Logo, a orientação completa para cada um dos $n$ vértices de $V$ irá consumir aproximadamente $n.2^n.n^{-1} = 2^n$ passos.

A complexidade do algoritmo de *Calabrese/França* cai drasticamente se trocamos as moedas equilibradas por moedas viciadas (*Calabrese/França polarizado*). Novamente, para grafos completos, tem-se que $\Pr\{moeda_i=1\}=1/n$ (considerando-se $n$-1 nós probabilísticos no passo). Assim:

$$\Pr(S) = \Pr\left(\bigcup_{i=1}^{n} S_i\right) = \sum_{i=1}^{n} \Pr(S_i) = \sum_{i=1}^{n} \left(\frac{1}{n}\right)\left(1-\frac{1}{n}\right)^{n-1} \geq \frac{n}{n.e} = \frac{1}{e}$$

Portanto, são esperadas $e$ repetições até que algum dos eventos $S_i$ (para $i \in \{1,..,n\}$) ocorra. Logo, para todos os nós tornarem-se determinísticos deverá ocorrer um número de passos aproximadamente igual a $n.e$, sendo portanto $O(n)$. Para maiores detalhes vide [CAL 94], [CAL 97].

## B. Alg-Viz

Esse algoritmo é uma extensão muito simples do anterior e propõe que o sorteio seja realizado com números em uma faixa de valores inteiros. Isso equivale à utilização de um dado de $f \geq 2$ faces (polarizado ou não) em vez de uma moeda. Assim, poderemos obter como resultado do sorteio números inteiros de 0 a $f$-1. Faremos inicialmente uma análise da correção e da complexidade esperada para dados não-polarizados introduzindo em seguida uma polarização. O caso polarizado entretanto será analisado apenas empiricamente, já que, para $f \geq n$, ele não irá representa uma redução na complexidade esperada quando comparado ao caso não-polarizado.

O *Alg-Viz* pode ser descrito da mesma forma que o algoritmo *Calabrese/França*, sendo necessário chamar a atenção apenas para o fato de que, agora, um nó ganha no sorteio quando tira o maior valor entre todos seus vizinhos probabilisticos.

O teorema seguinte garante correção do *Alg-Viz* e é válido igualmente para dados polarizados e não-polarizados. Obviamente, as análises de correção realizadas para o algoritmo *Alg-Viz* são igualmente válidas para *Calabrese/França*, já que, *Alg-Viz* apenas faz um particionamento maior na distribuição de probabilidades de *Calabrese/França*.

**Teorema 1.** O algoritmo *Alg-Viz* sempre gera orientações acíclicas em um grafo $G=(V,E)$ qualquer.

**Prova:** Vamos assumir que um nó, ao ganhar a disputa de dados com os vizinhos, recebe como rótulo um número $r$ inteiro positivo, que corresponde ao número de disputas de dados que ele participou até ser vitorioso. Sabemos que dois vizinhos não terão o mesmo rótulo, já que no passo que um ganha o outro obrigatoriamente não ganha. Após este processo, se algum nó $w$ ainda não rotulado não possuir vizinhos probabilísticos faremos simplesmente $r_w = \infty$. Assim, dados dois nós $u,v \in V$ e o seus rótulos $r_u$ e $r_v$, tais que, sem perda de generalidade, $r_u < r_v$, uma aresta $(u,v) \in E$ terá sempre a direção $v \rightarrow u$. Desta forma, o algoritmo corresponde a realizar uma coloração no grafo e orientar as arestas da maior para a menor cor o que certamente, não cria ciclos. □

### B.1 Complexidade Esperada

Como veremos a seguir, a utilização de dados em vez de moedas não modifica significativamente a complexidade da convergência a não ser que haja uma relação entre $f$ e uma característica global do sistema. A grande vantagem de *Alg-Viz* em relação a

*Calabrese/França* é diminuir (e, no caso, de dados com muitas faces, anular) o efeito dos empates entre vizinhos, incrementando, desta forma, a velocidade de convergência do algoritmo.

Mostraremos nesta seção que, para grafos completos com $n \geq 2$ (pior instância), o tempo esperado de processamento do *Alg-Viz* para dados equilibrados com $n$ faces será idêntico ao algoritmo *Calabrese/França* com moedas viciadas. Para isso, considere $d_i = \propto$ onde $\propto \in \{0,1,2,..,f\text{-}1\}$, o resultado associado a um nó $n_i \in V$, obtido após jogarmos um dado equilibrado com $f$ faces. Diremos que um evento $S_i$, associado ao nó $n_i \in V$ ocorre, se e somente se, $d_j < d_i$ , $\forall\ n_j \in N(n_i)$, onde $N(n_i)$ representa o conjunto dos vizinhos probabilísticos de $n_i$. Neste caso, o vértice $n_i \in V$ será chamado *nó campeão*. Observe que, se $G$ não é completo, teremos instâncias mais simples onde rodadas com mais de um nó campeão por iteração poderão implicar em uma convergência mais rápida do procedimento *Alg-Viz*. Em nossa análise consideramos apenas aquelas situações onde apenas um nó campeão ocorre de cada vez (pior caso).

Considere inicialmente o seguinte resultado válido para um grafo conexo $G$ qualquer:

**Teorema 2.** Considere $G=(V,E)$ um grafo conexo qualquer onde $|V| = n \geq 2$. Considere ainda um dado equilibrado com $f \geq 2$ faces e um evento $S_i$ que ocorre se $n_i \in V$ é campeão. Se $\Delta = max\{|N(n_j)|: n_j \in V\}$ então:

$$\Pr(S_i) \geq h(\Delta, f) \quad onde \quad h(\Delta, f) = \left(\frac{1}{\Delta+1}\right)\left(1-\frac{1}{f}\right)^{\Delta+1} + \frac{1}{2f}\left(1-\frac{1}{f}\right)^{\Delta} \quad \text{e} \quad n_i \in \mathrm{V}.$$

**Prova:** Da expressão de probabilidade absoluta (vide [JAM 81]) temos que:

$$\Pr(S_i) = \sum_{\alpha=0}^{f-1} \Pr(d_i = \alpha).\Pr(S_i \mid d_i = \alpha), \quad \forall\ n_i \in V \tag{I}$$

Associado a cada $n_i \in V$ com $d_i = \propto$ e $\propto \in \{0,1,..,f\text{-}1\}$, representaremos por $B_j(n_i, \propto)$, um evento que ocorre sempre que $d_j < d_i$ para algum $n_j \in N(n_i)$. Note que os eventos $B_j(n_i, \propto)$ para todo $n_j \in N(n_i)$ são independentes entre si. Além disso, temos $\Pr(B_j(n_i, \propto)) = \propto / f$. Assim:

$$\Pr(S_i \mid d_i = \alpha) = \Pr\left(\bigcap_{n_j \in N(n_i)} B_j(n_i, \alpha)\right) = \prod_{n_j \in N(n_i)} \left(\frac{\alpha}{f}\right) = \left(\frac{\alpha}{f}\right)^{|N(n_i)|}, \quad \forall\ i \in \{1,2,...,n\}. \tag{II}$$

Substituindo (II) em (I) e lembrando que $Pr(d_i = \propto) = 1/f$ obtemos:

$$\Pr(S_i) = \sum_{\alpha=0}^{f-1} \left(\frac{1}{f}\right)\left(\frac{\alpha}{f}\right)^{|N(n_i)|} \geq \sum_{\alpha=0}^{f-1} \left(\frac{1}{f}\right)\left(\frac{\alpha}{f}\right)^{\Delta}, \; p/\ \Delta = \max\{|N(n_i)|: n_i \in V\}$$

Logo:

$$\Pr(S_i) \geq \left(\frac{1}{f^{\Delta+1}}\right) \sum_{\alpha=0}^{f-1} \alpha^{\Delta} \tag{III}$$

Como $n \geq 2$ e $G$ é conexo então $\Delta \geq 1$. Um limite inferior para a série acima é descrito a seguir (para maiores detalhes vide [SPI 92]):

$$\sum_{\alpha=0}^{f-1} \alpha^{\Delta} \geq \left(\frac{(f-1)^{\Delta+1}}{\Delta+1} + \frac{(f-1)^{\Delta}}{2}\right) \tag{IV}$$

Finalmente, substituindo (IV) em (III), chegamos à expressão desejada. □

Seja $S$ um evento indicando a união de $n$ eventos distintos $S_i$, ou seja, $S$ ocorre se $S_i$ ocorre para algum $i \in \{1,..,n\}$. Cabe notar que, se G é um grafo qualquer, os eventos $S_i$, não são necessariamente disjuntos. Em outras palavras, poderemos ter mais de um nó campeão em uma mesma "jogada de dados".

O resultado seguinte (conseqüência do Teorema 2), mostra que, se $G$ é completo e o número de faces tende a infinito a probabilidade de orientação das arestas incidentes a algum $n_i \in V$, tende a 1. Em outras palavras, isto significa que o empate entre dados adjacentes tende a 0.

**Corolário 1:** Se $G$ é completo e $f \to \infty$ então $\Pr(S) \to 1$.

**Prova:** Como G é completo então $\Delta = n-1$ e $\Pr(S) = \Pr(\cup S_i) = \Sigma \Pr(S_i)$ (note neste caso que os eventos $S_i$ são disjuntos entre si). Fazendo $f \to \infty$ na expressão de $h(\Delta, f)$ (Teorema 2), temos $\Pr(S_i) \to 1/n$ e portanto $\Pr(S) \to 1$. □

**Teorema 3:** Considere $G=(V,E)$ um grafo completo onde $|V| = n \geq 2$. Considere ainda um dado equilibrado com $f \geq 2$ faces, sendo $f$ constante. O algoritmo *Alg-Viz* terá tempo esperado de processamento sub-exponencial e igual a:

$$O\left(f\left(\frac{f}{f-1}\right)^{n-1}\right)$$

**Prova:** Temos que $\Delta = n-1$ é grau máximo de G (já que $G$ é completo). Segue então do Teorema 2 que:

$$\Pr(S_i) \geq \left(\frac{1}{n}\right)\left(1-\frac{1}{f}\right)^n + \frac{1}{2f}\left(1-\frac{1}{f}\right)^{n-1} = \left[\left(\frac{1}{n}\right)\left(1-\frac{1}{f}\right)+\frac{1}{2f}\right]\left(1-\frac{1}{f}\right)^{n-1} = \left(\frac{2(f-1)+n}{2fn}\right)\left(1-\frac{1}{f}\right)^{n-1} \quad \text{onde} \quad n_i \in V.$$

Note agora que os eventos $S_i$ (para $i=1,2,\ldots,n$) tem interseção vazia (já que $G$ é completo). Segue então que:

$$\Pr(S) = \Pr\left(\bigcup_{i=1}^{n} S_i\right) = \sum_{i=1}^{n} \Pr(S_i) \geq \frac{1}{N} \quad onde \quad N = \left(\frac{2f}{2(f-1)+n}\right)\left(\frac{f}{f-1}\right)^{n-1}$$

Observe que $N$ repetições serão necessárias até que um nó campeão seja obtido (valor esperado). Em outras palavras, o número de repetições pode ser representado convenientemente por uma variável aleatória com distribuição geométrica e probabilidade de sucesso igual a $1/N$. Como $|V| = n$, conclui-se diretamente que o procedimento *Alg-Viz* terá tempo esperado igual a:

$$T(n) = \left(\frac{2fn}{2(f-1)+n}\right)\left(\frac{f}{f-1}\right)^{n-1} = O\left(f\left(\frac{f}{f-1}\right)^{n-1}\right)$$ □

**Corolário 2:** Se $G$ é completo e $f=n$ então $T(n)$ *é* $O(e.n)$.

**Prova:** Para provar este fato basta notar que: $(1-1/n)^{n-1} > 1/e$, $\forall\ n>1$. Equivalentemente, $(n/(n-1))^{n-1} < e$, $\forall\ n>1$. Fazendo $f=n$, obtemos do Teorema 2 que $T(n)$ é $O(e.n)$. □

Mostramos que, para grafos completos, o *Alg-Viz* para dados não-polarizados com $f$ constante, tem complexidade esperada sub-exponencial e pode ser visto como uma generalização natural do algoritmo de *Calabrese/França*[CAL 94] para moedas não-polarizadas. Mostramos ainda, que o algoritmo de *Calabrese/França*[CAL 94] para moedas polarizadas e o algoritmo *Alg-Viz* para dados com $n$ faces não-polarizados possuem

complexidade esperada $O(n.e)$. Neste caso entretanto, ao contrário do que ocorre com *Calabrese/França* [CAL 94], o desempenho do *Alg-Viz* pode ser melhorado, aumentando-se o número de faces do dado.

## B.2 Velocidade de Convergência

Da mesma forma que no algoritmo *Calabrese/França*, uma polarização dos dados no *Alg-Viz* tornará o algoritmo mais eficiente quando $f$ for constante. Através de testes realizados empiricamente, observamos que o comportamento do *Alg-Viz* cai de sub-exponencial para linear (Figura 1.a). Entretanto, se consideramos $f \geq n$, é fácil ver que, em ambas as versões do *Alg-Viz* (para dados polarizados e não-polarizados), o tempo esperado de processamento será sempre igual a $O(n)$. Isto se deve ao fato de que, em um grafo completo, apenas uma orientação na direção do nó campeão ocorre de cada vez (i.e, os eventos $S_i$ são disjuntos entre si). Contudo, testes os computacionais realizados indicam que a polarização introduzida melhora significativamente o desempenho do *Alg-Viz* quando $f$ aumenta com o tamanho do problema (Figura 1.b).

O método de polarização, descrito a seguir, é diferente para cada nó $n_i \in V$ do sistema e considera dados viciados com $f \geq 2$ faces.

$$\Pr\{d_i = f-1\} = \frac{1}{|N(n_i)|+1}$$

$$\Pr\{d_i = \alpha\} = \frac{1}{|N(n_i)|+1}\left(1 - \sum_{k=\alpha+1}^{f-1} \Pr(d_i = k)\right);\ p/\quad \alpha = f-2, f-3, \ldots, 2, 1$$

$$\Pr\{d_i = 0\} = 1 - \sum_{\alpha=1}^{f-1} \Pr(d_i = \alpha)$$

Desta forma, mantém-se uma polarização idêntica àquela utilizada em *Calabrese/França* para os casos em que $f = 2$. Esta polarização foi escolhida por ter se mostrado a mais eficiente na comparação com outras alternativas.

A Figura 1.a dá uma idéia, através de resultados de simulações, da melhora ocorrida com a utilização de polarizações quando $f$ é constante. No exemplo da Figura 1.a fizemos $f$=??. A Figura 1.b ilustra o desempenho da polarização para $f=n$. As simulações apresentadas nesse trabalho são versões seqüenciais dos algoritmos escritos em linguagem C. Cada ponto, de todos os gráficos produzidos, é a média aritmética obtida através da simulação de mil (1000) de grafos conexos gerados aleatoriamente.

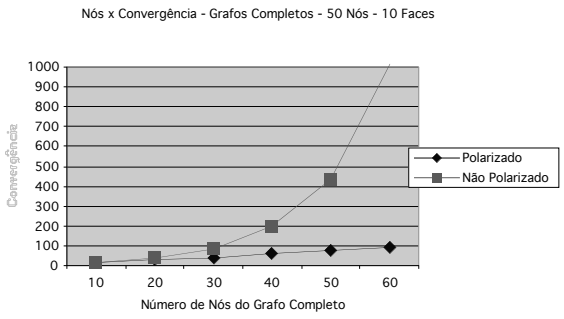


Figura 1.a: Alg-Viz. Dados polarizados e não-polarizados.

Como observado anteriormente, se $G$ é um grafo qualquer, o tempo esperado de processamento será inferior àquele obtido para grafos completos. Isto ocorre, já que *dois* ou mais nós campeões podem ser obtidos em uma mesma rodada (jogada de dados), acelerando a velocidade de convergência.

Antes de continuarmos com a análise empírica da velocidade de convergência do *Alg-Viz*, precisamos definir exatamente o que chamaremos de *densidade* de um grafo daqui para frente. Isso se deve ao fato de a densidade de um grafo, no sentido de "quão denso" é o grafo, exerce forte influência sobre o tempo de convergência, já que grafos mais densos tenderão a convergir mais devagar para o fim.

Para nós, a densidade $q$ de um grafo será a razão entre o número de arestas que o grafo possui e o número de arestas que ele poderia possuir com o mesmo número de nós. Logo

$$q = \frac{2m}{n(n-1)}.$$

Pela Figura 2, pode-se observar uma característica interessante acerca da velocidade de convergência do *Alg-Viz*. Utilizamos valores suficientemente grandes de $f$ ($f >> n$) de tal forma que este não prejudicasse a velocidade de convergência do algoritmo. Como discutido anteriormente, isto praticamente anula a possibilidade de empates nos sorteios. Pelo gráfico, claramente, temos que, mantendo-se $q$ constante, a velocidade de convergência é função linear de $n$. Além disso, o coeficiente da reta fica muito próximo de $q$.

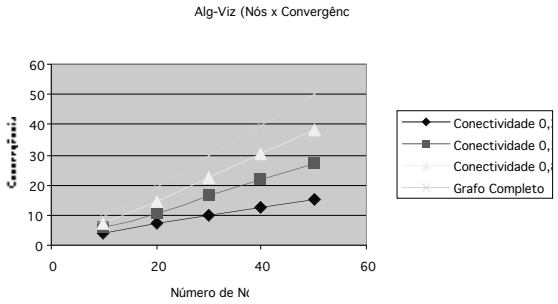


Figura 2: *Alg-Viz* (c/ dados polarizados)- com $q$ constante, a convergência é função linear de $n$.

## C. Alg-Arestas

Este algoritmo apresenta uma filosofia um pouco diferente dos anteriores (*Calabrese/França* e *Alg-Viz*). Em vez de tentar gerar sumidouros entre os nós probabilísticos, *Alg-Arestas* muda o foco de atenção para as arestas.

O *Alg-Arestas* também opera sincronamente e é descrito a seguir. Novamente, considere $d_i = \propto$ onde $\propto \in \{0,1,2,..,f\text{-}1\}$, o resultado associado a um nó $n_i \in V$, obtido após jogarmos um dado equilibrado com $f$ faces. Para cada aresta $(n_i,n_j) \in E$ (ainda não orientada), se $d_i > d_j$ então orientamos $(n_i,n_j)$ na direção de $n_i$ (aqui representada indistintamente por $n_i \leftarrow n_j$ ou $n_j \rightarrow n_i$). Se para alguma aresta tivermos $d_i = d_j$, repetimos novamente o processo até todas as arestas tenham sido orientadas.

Uma prova formal de que *Alg-Arestas* funciona é dada abaixo:

**Teorema 4.** *Alg-Arestas* sempre gera orientações acíclicas em qualquer grafo $G=(V,E)$.

**Prova.** Definimos $K$ como sendo o conjunto de todos os ciclos simples (sem nós repetidos) em $G=(V,E)$. A idéia é provar que o algoritmo não gera uma orientação cíclica para nenhum ciclo $\kappa \in K$. Sejam $n_{Y,0}$, $n_{Y,1}$,..., $n_{Y,|Y|-1}$, os vértices pertencentes a um ciclo $\kappa$ qualquer. Para simplificar a notação, considere simplesmente $n_{Y,i} = u_i$, para todo $0 \leq i \leq |\kappa|\text{-}1$. Suponha ainda que os vértices $u_0,...,u_{|\kappa|-1}$, sejam selecionados de maneira que $u_i$ esteja conectado por uma aresta de $\kappa$ a $u_{(i+1) \bmod |\kappa|}$, $0 \leq i \leq |\kappa|\text{-}1$. Os números sorteados para os nós $u_i$ são dados por $d^{u}{}_{i}$. Temos duas possibilidades:

(a) $d^u_0=d^u_1=...=d^u_{|\kappa|-1}$. Nesse caso, nenhuma aresta do ciclo é orientada e o sorteio é executado novamente;

(b) $\exists\ a,\ 0 \le a \le |\kappa|-1$, tal que $d^u_a < d^s_{(a+1)\bmod|\kappa|}$. Nesse caso, tomemos uma nova nomeação dos nós como $v_0,v_1,...,v_{|\kappa|-1}$ tal que $v_i=u_{(a+i)\bmod|\kappa|}$ , para $i$=0,1,..., $|\kappa|$-1. Como $v_0=u_a$ e $v_1=u_{(a+1)\bmod|\kappa|}$, já sabemos que $d^v_0 < d^v_1$, portanto a aresta $(v_0,v_1)$ terá a direção $v_0 \rightarrow v_1$. Desta forma, se $\forall\ j,\ 1 \le j \le |\kappa|-2$, tivermos que $d^v_j \le d^v_{j+1}$, então $d^v_{|\kappa|-1} > d^v_0$ e, portanto, a aresta $(v_{|\kappa|-1}, v_0)$ será orientada na direção $v_0 \rightarrow v_{|\kappa|-1}$, impossibilitando a formação de uma orientação cíclica em $\kappa$ nos passos seguintes do algoritmo. Por outro lado, se $\exists\ j,\ 1 \le j \le |\kappa|-2$ tal que $d^v_j > d^v_{j+1}$, então a aresta $(v_j,v_{j+1})$ será orientada na direção $v_j \leftarrow v_{j+1}$. Assim, teremos a seguinte situação (o símbolo $\leftrightarrow$ significa orientação indefinida) que define a impossibilidade de formação de uma orientação cíclica em $\kappa$ nos passos seguintes do algoritmo.

$$v_{|\kappa|-1} \leftrightarrow v_0 \rightarrow v_1 \leftrightarrow \ldots \leftrightarrow v_j \leftarrow v_{j+1} \leftrightarrow \ldots \leftrightarrow v_{|\kappa|-1}. \qquad \square$$

## C.1 Complexidade Esperada

É bastante fácil perceber que *Alg-Arestas* tem convergência muito mais rápida que *Alg-Viz*. Nesta seção provaremos que o tempo esperado de convergência será inferior a *O(n)* mesmo considerando-se grafos completos.

A idéia da prova de complexidade consiste no seguinte: em um sorteio, a probabilidade de empate entre dois vizinhos é de 1/*f*. Assim, enquanto o número de arestas é grande, esta é a proporção de empates ocorridos a cada passo do algoritmo. Logo, em cada passo, uma proporção de 1/*f* das arestas não orientadas até aquele passo permanecem sem serem orientadas. Portanto, a convergência é dada por $O(\lceil \log_f m \rceil)$, onde $m$ é o número de arestas do grafo original $G=(V,E)$. Apesar de intuitivamente razoável e de apresentar resultados próximos da realidade, esta análise é dificultada já que a proporção 1/*f* só é válida quando o número de arestas sendo consideradas é suficientemente grande.

Mais formalmente, este processo pode ser descrito por uma relação de recorrência do tipo $T(x)=a(x)+T(h(x))$, onde $x$ é variável real não-negativa, $a(x)$ é função real não-decrescente e não-negativa de $x$, e $h(x)$ é variável aleatória assumindo valores no intervalo $[0,x]$. Além disso, considere uma função real $g(x)$ onde $E(h(x)) \le g(x) \le x$, sendo $g(x)$ e $g(x)/x$ funções não-decrescentes e não-negativas de $x$ [KAR 94]. A abordagem de [KAR 94] é bastante genérica já que, não se exige, a priori, nenhuma informação sobre a distribuição de $h(x)$. Note que $a(x)$ representa o esforço necessário para se quebrar o problema original em um subproblema de tamanho $h(x)$.

A equação $\tau(x)=a(x)+\tau(g(x))$, pode ser vista como uma versão determinística da relação de recorrência probabilística $T(x)$. Neste caso, para todo $x$, a variável aleatória $h(x)$ será exatamente igual ao limite superior de seu valor esperado, ou seja, faremos $E(h(x))=g(x)$. Como discutido em [KAR 94], o *Teorema do Ponto-Fixo de Tarski* garante que, sempre que esta equação tem solução não-negativa, existe uma única solução (não-negativa) $u(x)$ onde $u(x) \le v(x),\ \forall\ x$, e qualquer que seja a solução $v(x)$ não-negativa. Em outras palavras, $u(x)$ representa a menor solução não-negativa de $\tau(x)=a(x)+\tau(g(x))$. A função $u(x)$ é dada explicitamente pela fórmula:

$$u(x) = \sum_{i=0}^{\infty} a\left(g^{[i]}(x)\right)$$

onde $g^{[i]}(x)$ é definido indutivamente por $g^{[i]}(x)= g(g^{[i-1]}(x))$ para $i$=1,2,.. e $g^{[0]}(x)=x$.

Temos então o seguinte resultado provado em [KAR 94]:

**Teorema 5.** Suponha que exista uma constante $d$ tal que $a(x)=0,\ x<d$ e $a(x)=1,\ x \ge d$. Seja $c_t = \min\{x | u(x) \ge t\}$. Então, para todo real positivo $x$ e todo inteiro positivo $w$,

$$\Pr[T(x) \geq u(x) + w] \leq \left(\frac{g(x)}{x}\right)^{w-1} \frac{g(x)}{c_{u(x)}}.$$

Note, da proposição acima que, se $u(x)$ representa o tempo esperado de processamento então $Pr[T(x) \geq u(x)+w]$ indica a probabilidade de nos afastarmos de $u(x)+w$ (onde $w$ é inteiro positivo). Este resultado é interessante pois nos dá uma medida do grau de variação de $T(x)$ em torno de seu valor esperado. O Teorema 5, como veremos a seguir, pode ser utilizado na prova da complexidade esperada do procedimento *Alg-Arestas*. Temos então o seguinte resultado:

**Teorema 6.** Dado um grafo $G=(V,E)$ com $m$ arestas e um dado equilibrado com $f$ faces, o algoritmo *Alg-Arestas* tem tempo esperado de processamento $T(m)$ igual a $O(\lfloor \log_f m \rfloor +1)$. Além disso:

$$\Pr[T(m) \geq \lfloor \log_f m \rfloor + 1 + w] \leq \left(\frac{1}{f}\right)^{w-1} \frac{m}{f^{\lfloor \log_f m \rfloor + 1}}$$

**Prova.** Observe em nosso caso que, fazendo-se $x=m$ e $d=1$, temos $a(m)=1$ para $m \geq 1$ e $a(m)=0$ para $m<1$. Além disso, $h(m)$ é variável aleatória indicando quantidade de arestas resultantes não orientadas após a primeira iteração.

Cada aresta é orientada independentemente das outras. Portanto, podemos definir uma variável aleatória de Bernoulli que modela cada tentativa de orientação de uma aresta, ou seja, a disputa de dados entre os dois nós que são conectados por esta. A probabilidade de fracasso, ou seja, de a aresta não se orientar, é igual à de dar empate na disputa entre dois dados equilibrados de número de faces iguais a $f$, ou seja, $1/f$. Assim, $1/f$ representa a probabilidade de fracasso e $1-1/f$ a probabilidade de sucesso.

Note que número total de arestas orientadas a cada iteração (somatório dos ensaios de Bernoulli) pode ser representado por uma variável aleatória com distribuição binomial e valor esperado $m(1-1/f)$. Logo, $g(m) = m/f$, irá representar o número de arestas não orientadas após a primeira iteração. Além disso, temos que $E[h(m)]=g(m)$, $g(m)$ e $g(m)/m$ são funções não-decrescentes em $m$. A solução $u(m)$ de $\tau(m)=a(m)+\tau(g(m))$ pode então ser obtida iterando-se $u(m)=1+u(m/f)$ $k$ vezes obtendo $u(m)=k+ u(m/f^k)$. O processo é repetido até que $m/f^k=1$, ou seja, $k= \log_f m$. Como $k$ é inteiro fazemos $k=\lfloor \log_f m \rfloor$. Finalmente, como $u(1)=1$, obtemos $u(m)= \lfloor \log_f m \rfloor +1$.

Para provar a segunda parte, mostraremos agora que $c_t = f^{t-1}$. De fato, $c_t = \min\{m | u(m) \geq t\}$ (Teorema 5) . Assim:

$$u(m) \geq t \Rightarrow \lfloor \log_f m \rfloor + 1 \geq t \Rightarrow \log_f m \geq t-1 \Rightarrow f^{\log_f m} \geq f^{t-1}$$

Portanto, temos que: $c_t = \min\{m \mid m \geq f^{t-1}\} = f^{t-1}$. Finalmente, substituindo $u(m)$, $c_{u(m)}$ e $g(m) = m/f$ em:

$$\Pr[T(m) \geq u(m) + w] \leq \left(\frac{g(m)}{m}\right)^{w-1} \cdot \frac{g(m)}{c_{u(m)}}$$

obtemos:

$$\Pr[T(m) \geq \lfloor \log_f m \rfloor + 1 + w] \leq \left(\frac{1}{f}\right)^{w-1} \frac{m}{f^{\lfloor \log_f m \rfloor + 1}}$$

como queríamos demonstrar. □

Para ilustrar esse resultado, note que, se $m = f^k$ para algum $k \in Z^+$ e $w$=1 então: $\Pr[T(m) \geq u(m)+1] \leq 1/f$. Este resultado é interessante pois mostra que a probabilidade de nos afastarmos apenas uma unidade do valor esperado independe do número de arestas em $G$ e é inversamente proporcional ao número de faces do dado.

Note ainda que, mesmo para grafos completos (onde $m=O(n^2)$), o desempenho do *Alg-Arestas* é significativamente superior ao *Alg-Viz*. O tempo de convergência de *Alg-Arestas* é bastante pequeno quando comparado com *Alg-Viz*. Na verdade, na situação em que $f>>m$, podemos considerar que o algoritmo converge em apenas um (1) passo, o que é um resultado muito bom, já que podemos sem muito custo utilizar valores bastante grandes de $f$.

A Tabela I exibe, para alguns valores de $m$ e $f$, os resultados obtidos por simulação (média aritmética obtida através da simulação de mil (1000) de grafos conexos gerados aleatoriamente), pela expressão aproximada $\lceil \log_f m \rceil$.

TABELA I

COMPARAÇÃO ENTRE A VELOCIDADE ESPERADA DE CONVERGÊNCIA E AS OBTIDAS NAS SIMULAÇÕES

| Arestas | Faces | Simulação | $\lceil \log_f m \rceil$ |
|---|---|---|---|
| 100 | 10 | 2.70 | 2 |
| 200 | 110 | 1.82 | 2 |
| 300 | 210 | 1.79 | 2 |
| 400 | 310 | 1.72 | 2 |
| 500 | 410 | 1.70 | 2 |

Na Figura 3 adiante podemos observar o tempo de convergência do algoritmo para diferentes valores de $f$ e $m$. Note que os valores são realmente pequenos, mesmo para um número reduzido de faces do dado.

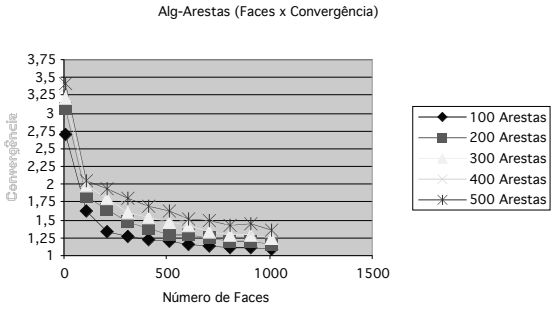


Figura 3: Velocidade de convergência de *Alg-Arestas.*

## IV. APLICAÇÕES ALVO

Nesta seção estaremos analisando duas aplicações onde a definição de uma orientação acíclica pode ser um passo frequentemente requerido durante o processamento da aplicação ou sua correta inicialização. A primeira aplicação é sobre o problema do escalonamento

distribuído de recursos entre processos em sistemas distribuídos. Uma segunda aplicação versa sobre uma estratégia de distribuição de *uploads* em redes de computadores.

## A. Escalonamento por Reversão de Arestas

O *Escalonamento por Reversão de Arestas* [BAR 89] [BAR 96] visa resolver o problema de controlar o acesso a recursos compartilhados entre processadores em um ambiente distribuído. Assim, dado um conjunto $V$ de nós de processamento, um conjunto $R$ de recursos e um conjunto $D \subseteq (V \times R)$ dos pares $(n \in V, r \in R)$ em que o nó $n$ utiliza o recurso $r$, determinar a ordem de execução (ou operação) dos nós de tal forma que dois nós que compartilham o mesmo recurso não executem ao mesmo tempo e de tal forma que não ocorra *deadlock* nem *starvation*. O ERA foi criado para condições de alta carga no sistema, ou seja, cada nó está sempre precisando de todos os recursos compartilhados ao seu alcance para que possa operar.

A partir dessa definição podemos construir um *Grafo de Dependências* $G^D=(V,E^D)$ onde $E^D \subseteq E$, que irá representar os padrões de compartilhamento do sistema. O conjunto $E^D$ é composto das arestas que ligam dois nós que compartilham pelo menos um recurso, ou seja, $m = (n_i,n_j) \in E^D$ se, e somente se, $(n_i,n_j) \in E$ e $\exists\ r \in R$, tal que, $(n_i,r) \in D$ e $(n_j,r) \in D$. Observe que, nessa definição, assumimos que um recurso só poderá ser compartilhado entre nós que tenham um canal de comunicação entre si, o que é razoável. Também vale observar que, nessa notação, cada recurso compartilhado corresponde a um subgrafo completamente conectado, já que todos os nós que o compartilham terão arestas entre eles.

O algoritmo ERA admite a existência inicial de uma orientação acíclica no grafo $G^D$. Basicamente uma aresta $(n_i,n_j)$ orientada no sentido $n_i \rightarrow n_j$ indica que o nó $n_j$ tem precedência em relação ao nó $n_i$ na próxima operação. A necessidade de a orientação ser acíclica é facilmente percebida, já que, de acordo com a definição acima, um ciclo implicaria uma cadeia de precedências entre nós organizada de forma a gerar um bloqueio perpétuo ou *deadlock*.

Assim, um nó para operar precisa ser um *sumidouro*, ou seja, deverá ter todas as arestas orientadas em sua direção, já que isso equivale a ter precedência sobre todos os vizinhos (como já foi dito, vizinhos em $G^D$ não podem operar ao mesmo tempo). Uma orientação acíclica em $G^D$ implica a existência de pelo menos um sumidouro em $V$, assim é garantido que o algoritmo pode sempre iniciar com os sumidouros operando. Após sua operação, cada sumidouro reverte as suas arestas na direção oposta. A nova orientação gerada é também acíclica, pois todos os sumidouros foram transformados em fontes e esse tipo de operação claramente não cria ciclos. Por isso, um novo conjunto de sumidouros será obtido. O processo, então se repete, com os sumidouros operando e revertendo suas arestas, definindo assim o escalonamento pretendido.

## B. Upload Distribuído

Nesse problema, visamos contornar o congestionamento natural no tráfego de informações em uma determinada rede de comunicação, na situação particular onde um único ponto de recepção é responsável pela captação de todas as informações provenientes de todos os pontos dessa rede. Esta é uma situação bastante comum, onde uma data limite para entrega de dados por toda uma população específica é estabelecida e o fluxo de dados se intensifica nas proximidades dessa data provocando o congestionamento dos canais de comunicação, e.g., entrega da declaração do imposto de renda.

Uma alternativa viável para atenuar esse quadro estaria na multiplicação de nós receptores na rede. O problema do *upload distribuído* consiste na determinação do escoamento apropriado dos dados em direção aos nós receptores. Uma abordagem possível estaria na geração de uma orientação acíclica sobre o grafo $G$ representando a rede de comunicações alvo, onde os nós de $G$ seriam os roteadores e as arestas de $G$ os canais de comunicação existentes. Nessa orientação acíclica, feita sobre as arestas de $G$, os roteadores associados aos pontos de recepção corresponderiam aos únicos nós sumidouros que assumiriam, localmente, a distância

(a qualquer sumidouro) $d = 0$. Para se obter tal configuração orientar-se-ia inicialmente as arestas adjecentes aos nós receptores de forma a definir o conjunto de sumidouros desejado (desprezando-se possíveis arestas entre nós receptores vizinhos). Os nós restantes assumiriam as respectivas menores distâncias a qualquer sumidouro e a orientação das arestas entre nós de diferentes distâncias seguiria a ordem decrescente. A seguir, o uso de qualquer dos algoritmos para geração de orientações acíclicas poderia ser utilizado entre nós de igual distância em *G*.

## V. CONCLUSÕES

Foram apresentados e analisados dois algoritmos distribuídos probabilísticos para geração de orientações acíclicas em sistemas distribuídos anônimos: *Alg-Viz* (uma extensão direta do algoritmo *Calabrese/França*) e *Alg-Arestas* (que, em certas circunstâncias, gera orientações acíclicas em menos de 2 passos).

Provamos a correção do *Alg-Viz* e *Alg-Arestas*. Além disso, a análise de seus tempos esperados de processamento foi realizada, tanto através de simulações quanto teoricamente.

Apresentamos ainda duas aplicações onde a definição de uma orientação acíclica era passo essencial no processamento da aplicação. Na primeira aplicação discutiu-se o problema do escalonamento distribuído de recursos entre processos em sistemas distribuídos. Na segunda aplicação esboçamos uma estratégia de distribuição de *uploads* em redes de computadores.

Um possível trabalho futuro é a utilização dos algoritmos apresentados para gerar orientações em multigrafos como passo inicial para *SMER* (*Scheduling by Multiple Edges Reversal*) [BAR 01], uma generalização de ERA para sistemas heterogêneos no tocante ao regime de carga dos processos.

## Agradecimentos

Agradecemos aos revisores deste trabalho pelas valiosas sugestões e comentários.

## REFERÊNCIAS

[ARA 99] Arantes Jr., G. M., (1999). *Orientações Acíclicas em Sistemas Distribuídos Anônimos e suas Aplicações no Compartilhamento de Recursos*. Tese de M.Sc., COPPE/UFRJ.

[ARA 01] Arantes Jr, G. M. & França, F.M.G., (2001). Geração Quase-Instantânea de Orientações Acíclicas em Sistemas Distribuídos Anônimos. *Anais do II Workshop em Sistemas Computacionais de Alto Desempenho*, Pirenópolis, GO, pp. 55-62, Set.

[BAR 01] Barbosa, V. C., Benevides, M. R. F., França, F. M. G. (2001). Sharing resources at nonuniform access rates. *Theory of Computing Systems*, vol. 34, Springer, pp. 13-26.

[BAR 89] Barbosa, V. C. (1989). Concurrency in Heavily Loaded Neighborhood-Constrained Systems. *ACM Transactions on Programming Languages and Systems*, Vol. 11, n. 4, pp. 562-584.

[BAR 96] Barbosa, V. C. (1994). An Introduction to Distributed Algorithms, MIT Press 1996.

[CAL 94] Calabrese, A., & França, F. M. G. (1994). A Randomized Distributed Primer for the Updating Control of Anonymous ANNs. *Proceedings of the International Conference on Artificial Neural Networks 94*, Sorrento, Italy. **(PAGINAS??)**

[CAL 97] Calabrese, A. (1997). Distributed Acyclic Orientation of Asynchronous Networks, *11th International Symposium of Fundamentals of Computation Theory*, pp.129-137, Kraków, Poland.

[FRA 91] França, F. M. G. (1991). A Self-Organizing Updating Network, In *Artificial Neural Networks,* eds. T. Kohonen, K, Mckisara, O. Simula, and J. Kangas, Elsevier Science Publisher B. V. (North-Holland), pp. 1349-1352.

[FAR 99] Faria, L. & França, F. M. G. (1999). Comunicação Pessoal.

[GOL 87] Goldberg, A. V. & Plotkin, S. A. (1987). Parallel ($\Delta$+1)-Coloring of Constant-Dregree Graphs. *Information Processing Letters*, n. 25, pp. 241-245.

[GOL 88] Goldberg, A. V. & Plotkin, S. A., (1988). Parallel Symmetry-Breaking in Sparse Graphs. *SIAM Journal of Discrete Mathematics*, Vol. 1, n .4, pp. 434-445.

[ITA 90] Itai, A. & Rodeh, M., (1990). Symmetry-Breaking in Distributed Networks. *Information and Control*, n. 88, pp. 60-87.

[JAM 81] James, B.R., (1981). *Probabilidade: um curso em nível intermediário*. Projeto Euclides.

[KAR 94] Karp R., (1994). Probabilistic Recurrence Relations. *Journal of ACM*. Vol 41. No 6, pp 1100-1136.

[LUB 86] Luby, M., (1986). A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM Journal of Computing*, Vol. 15, n. 4, pp. 1036-1053.

[PAN 92] Panconesi, A. & Srinavasan, A., (1992). Improved Distributed Algorithms for Coloring and Network Decomposition Problems. *24th ACM Symposium on Theory of Computation*, pp. 581-591, Victoria B. C., Canada.

[SPI 92] Spiegel, M. R., (1992). *Manual de Fórmulas, Métodos e Tabelas de Matemática*. Makron, McGraw-Hill (Coleção Schaum).

[SZE 93] Szegedy, M. & Vishwanathan S., (1993). Locality Based Graph Coloring. *25th ACM Symposium on Theory of Computation*, pp. 201-207, California, USA.