versão impressa ISSN 0101-7438 / versão online ISSN 1678-5142

GERANDO ORIENTAÇÕES ACÍCLICAS COM ALGORITMOS PROBABILÍSTICOS DISTRIBUÍDOS

Gladstone M. Arantes Jr.
Felipe M. G. França
Progr. de Eng. de Sistemas e Computação / COPPE
Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ
glads@cos.ufrj.br
felipe@cos.ufrj.br

Carlos A. Martinhon *
Instituto de Computação
Universidade Federal Fluminense (UFF)
Niterói – RJ
mart@dcc.ic.uff.br

* *Corresponding author* / autor para quem as correspondências devem ser encaminhadas

Recebido em 03/2003; aceito em 08/2005 após 1 revisão
Received March 2003; accepted August 2005 after one revision

Resumo

Este artigo apresenta um novo algoritmo distribuído probabilístico para a geração de orientações acíclicas em um sistema distribuído anônimo de topologia arbitrária. O algoritmo é analisado tanto em termos de correção e complexidade esperada quanto velocidade de convergência. Em particular, é demonstrado que este novo algoritmo, chamado *Alg-Arestas*, é capaz de produzir, com alta probabilidade, orientações acíclicas quase instantaneamente, isto é, em menos de dois passos. Duas aplicações para essa forma de quebra de simetria serão discutidas: (i) inicialização do *Escalonamento por Reversão de Arestas* (*ERA*), um simples e poderoso algoritmo de escalonamento distribuído, e (ii) uma estratégia de distribuição de *uploads* em redes de computadores.

Palavras-chave: algoritmos distribuídos probabilísticos; quebra de simetria; sistemas anônimos.

Abstract

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

Keywords: randomized distributed algorithms; symmetry breaking; anonymous systems.

1. Introdução

Neste trabalho apresentaremos um novo algoritmo probabilístico que gera 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 seqüenciais, 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 mais adiante.

Estudaremos correção, complexidade esperada e velocidade de convergência do novo algoritmo probabilístico distribuído (aqui denominado *Alg-Arestas*). De maneira geral, nos algoritmos probabilísticos, para 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 ser expresso como uma *variável aleatória* com *valor esperado* indicando sua complexidade esperada. O *Alg-Arestas* faz o uso de *dados*, não-viciados, com $f \geq 2$ faces. A análise deste novo algoritmo foi baseada na noção de *recorrência probabilística* [KAR 94] e resultou em complexidade esperada igual a $O(\log_f m)$ para um grafo qualquer com $|E|=m$ arestas e $f \geq 2$. Este resultado é superior ao *Alg-Viz* (apresentado em [ARA 02]) cuja complexidade esperada é igual a $O(n)$ para dados viciados convenientemente.

Na Seção 2, introduzimos conceitos básicos: sistemas distribuídos anônimos e sua representação em termos de grafos. Na Seção 3, apresentamos os algoritmos básicos de *Calabrese/França* [CAL 94] [CAL 97] e introduzimos um novo algoritmo de geração de orientações acíclicas (*Alg-Arestas*), discutindo sua correção, complexidade esperada e velocidade de convergência. Algumas aplicações envolvendo a determinação de orientações acíclicas no contexto de sistemas anônimos são analisadas na Seção 4. Finalmente, nossas conclusões, incluindo futuros desenvolvimentos, são expostas na Seção 5.

2. Conceitos Básicos

2.1 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. Normalmente, 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].

2.2 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.

3. Os Algoritmos Distribuídos

Descrevemos e analisamos aqui um novo algoritmo distribuído probabilístico de nosso interesse (*Alg-Arestas*). Este novo algoritmo foi desenvolvido a partir das idéias básicas contidas em [CAL 94], [CAL 97] e [ARA 02], traduzidas pelos algoritmos de *Calabrese/ França* e *Alg-Viz*, ambos nas versões *não-polarizada* e *polarizada.* Com o único fim de facilitar o entendimento de todos os três algoritmos distribuídos, que serão apresentados a seguir, será assumido um comportamento síncrono dos mesmos, o que não é necessário ao correto funcionamento dos mesmos. Na verdade, em muitos casos interessantes, como nos exemplos de aplicações oferecidos ao final deste artigo, os ambientes alvos são assíncronos.

3.1 *Calabrese*/*França* e *Alg-Viz*

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*.

Considere que 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.

Como observado mais adiante, o algoritmo de *Calabrese/França* não-polarizado é bastante ineficiente. Por isso, em sua versão polarizada (que também opera sincronamente) as moedas são *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 falando, em uma dada vizinhança apenas um dos nós tende a obter 1 no sorteio.

3.1.1 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 índice $i \in \{1,2,..,n\}$) ocorrerá de cada vez, ou seja, $S_i \cap S_j = \varnothing$ para $i \neq j$. 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 (onde $e \cong 2{,}17$) 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].

O algoritmo *Alg-Viz* é uma extensão muito simples do anterior e propõe que o sorteio seja realizado com números em uma faixa maior de valores inteiros. Isso equivale à utilização de um dado de $f \geq 2$ faces (polarizado ou não) em vez de uma moeda. Assim, pode-se obter como resultado do sorteio números inteiros variando de 0 a f-1.

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 (*nó campeão*) quando tira o maior valor entre todos seus vizinhos probabilísticos. Em [ARA 02] é demonstrado que o *Alg-Viz* para dados não-polarizados e f=n possui complexidade esperada igual a $O(n)$, e portanto, igual ao Algoritmo de *Calabrese/França* para moedas polarizadas. A grande vantagem do *Alg-Viz* em relação ao procedimento de *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.

Como discutido em [ARA 02], da mesma forma que no algoritmo *Calabrese/França*, uma polarização (realizada convenientemente) de dados com mais de 2 faces tornará o *Alg-Viz* mais eficiente. Neste caso entretanto, 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. Desta forma, isto impede que se tenha ganhos significativos no *Alg-Viz* introduzindo-se uma polarização para dados com f>2. Maiores detalhes sobre o *Alg-Viz* podem ser encontrados em [ARA 02].

3.2 *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 de G.

Assuma uma operação síncrona do *Alg-Arestas*. 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 a seguir:

Teorema 1. *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 $c\in K$. Sejam $n_{c,0}$, $n_{c,1}$,..., $n_{c,|c|\text{-}1}$, os vértices pertencentes a um ciclo c qualquer. Para simplificar a notação, considere simplesmente $n_{c,i} = u_i$, para todo $0 \le i \le |c|\text{-}1$. Suponha ainda que os vértices $u_0,...,u_{|c|\text{-}1}$, sejam selecionados de maneira que u_i esteja conectado por uma aresta de c a $u_{(i+1) \bmod |c|}$, $0 \le i \le |c|\text{-}1$. Os números sorteados para os nós u_i são dados por d^u_i. Temos duas possibilidades:

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

(b) $\exists\ a$, $0 \le a \le |c|\text{-}1$, tal que $d^u_a < d^u_{(a+1)\bmod|c|}$. Nesse caso, tomemos uma nova nomeação dos nós como $v_0,v_1,...,v_{|c|\text{-}1}$ tal que $v_i=u_{(a+i)\bmod|c|}$, para i=0,1,..., $|c|$-1. Como $v_0=u_a$ e $v_1= u_{(a+1)\bmod|c|}$, 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 $\forall\ j$, $1\le j\le|c|\text{-}2$, tivermos que $d^v_j \le d^v_{j+1}$, então $d^v_{|c|\text{-}1}> d^v_0$ e, portanto, a aresta $[v_{|c|\text{-}1},v_0]$ será orientada na direção $v_0\rightarrow v_{|c|\text{-}1}$, impossibilitando a formação de uma orientação cíclica em c nos passos seguintes do algoritmo. Por outro lado, se $\exists\ j$, $1 \le j \le |c|\text{-}2$ 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 c nos passos seguintes do algoritmo:

$$v_{|c|\text{-}1} \leftrightarrow v_0 \rightarrow v_1 \leftrightarrow ... \leftrightarrow v_j \leftarrow v_{j+1} \leftrightarrow ... \leftrightarrow v_{|c|\text{-}1}.$$

3.2.1 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 $O(\log_f m)$, 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 $[0,x]$ (em nosso caso, a variável x representa as arestas ainda não orientadas em G). Considere ainda 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 2. 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,

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

Note, da proposição acima que, se $u(x)$ representa o tempo esperado de processamento então $Pr[T(x)\ge u(x)+w]$ 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 3, como veremos a seguir, pode ser utilizado na prova da complexidade esperada do procedimento *Alg-Arestas*. Temos então o seguinte resultado:

Teorema 3. 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 $\lfloor \log_f m \rfloor +1$. Além disso:

$$\Pr[T(m) \geq \lfloor \log_f m \rfloor + 1 + w] \leq \left(\frac{1}{f}\right)^{w-1} \frac{m}{f^{\lfloor \log_f m \rfloor + 1}} \qquad (2)$$

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 2) . 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 (1) 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 de (2) 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* passo, o que é um resultado muito bom, já que podemos sem muito custo utilizar valores bastante grandes de f.

A Tabela 1 exibe, para alguns valores de m e f, os resultados obtidos por simulação (média aritmética obtida através da simulação de 1000 grafos conexos gerados aleatoriamente), pela expressão aproximada $\lfloor \log_f m \rfloor+1$.

Tabela 1 – Comparação entre a velocidade esperada de convergência e as obtidas nas simulações.

Arestas	Faces	Simulação	$\lfloor \log_f m \rfloor + 1$
100	10	2.70	3
200	110	1.82	2
300	210	1.79	2
400	310	1.72	2
500	410	1.70	2

Na Figura 1 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.

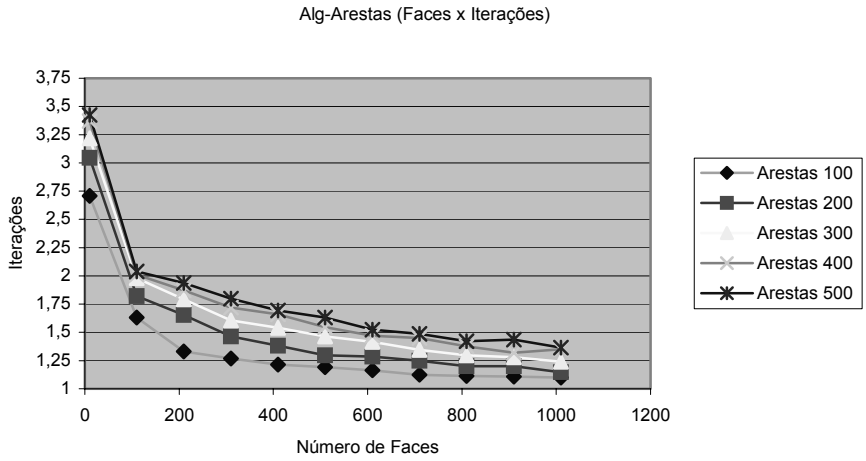

Figura 1 – Velocidade de convergência de *Alg-Arestas*.

4. Aplicações Alvo

Nesta seção estaremos analisando duas aplicações onde a definição de uma orientação acíclica pode ser um passo freqüentemente 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.

4.1 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.

4.2 *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. Outra situação interessante, que também pode ser atacada de forma análoga, seria a de se lidar com múltiplas transmissões do tipo *multicast* simultaneamente, e.g., roteamento de tráfego de rede advindo de diversas classes concorrentes em um sistema de educação à distância.

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 adjacentes 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. Finalmente, far-se-ia uso do algoritmo *Alg-Arestas* para uma rápida geração de orientações acíclicas entre todos os nós vizinhos com igual distância d em G.

5. Conclusões

Foi apresentado e analisado um novo algoritmo distribuído probabilístico para geração de orientações acíclicas em sistemas distribuídos anônimos: *Alg-Arestas* (que, em certas circunstâncias, gera orientações acíclicas em menos de 2 passos). Provamos a correção do *Alg-Arestas* e fizemos uma análise de seu tempo esperado de processamento, análise esta, 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 compartilhados entre processos em sistemas distribuídos. Na segunda aplicação esboçamos uma estratégia de distribuição de *uploads* em redes de computadores. Note que a mesma estratégia poderia ser aplicada em um esquema de *download* distribuído para os casos em que os arquivos alvo estivessem particionados e distribuídos pela rede. Nesse caso, os nós sumidouros reagrupariam as partes dos arquivos alvo. É interessante observar ainda que toda orientação acíclica está necessariamente associada a uma coloração de G dado que, através de uma *decomposição por sumidouros* [STA 76], i.e., uma organização dos nós de G pela maior distância, via caminhos orientados, até um sumidouro qualquer. Todos os nós que possuírem uma mesma distância, também podem possuir uma mesma cor dado que serão não-vizinhos, necessariamente.

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 ao CNPq, à CAPES pelo suporte e aos revisores deste trabalho pelas valiosas sugestões e comentários.

Referências Bibliográficas

[ARA 99] Arantes Jr., G.M. (1999). Orientações Acíclicas em Sistemas Distribuídos Anônimos e suas Aplicações no Compartilhamento de Recursos. Dissertação 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, 55-62.

[ARA 02] Arantes Jr, G.M.; França, F.M.G. & Martinhon C.A. (2002). Algoritmos Randômicos na Geração de Orientações Acíclicas em Sistemas Distribuídos. *Anais do XXXIV Simpósio Brasileiro de Pesquisa Operacional*, Rio de Janeiro, RJ.

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

[BAR 89] Barbosa, V.C. & Gafni, E. (1989). Concurrency in Heavily Loaded Neighborhood-Constrained Systems. *ACM Transactions on Programming Languages and Systems*, **11**(4), 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, 585-588.

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

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

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

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

[ITA 90] Itai, A. & Rodeh, M. (1990). Symmetry-Breaking in Distributed Networks. *Information and Control*, **88**, 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*, **41**(6), 1100-1136.

[LUB 86] Luby, M. (1986). A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM Journal of Computing*, **15**(4), 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*, Victoria B. C., Canada, 581-591.

[STA 76] Stahl, S. (1976). *n*-tuple colorings and associated graphs. *J. Combinatorial Theory*, Ser. B, **20**(2), 185-203.

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