Um Algoritmo de Geração Sequencial de Estados numa Rede Sujeita a Avarias

por

Teresa Martínez Gomes

José Fernandes Craveirinha

INESC – Núcleo de Coimbra

Relatório de Investigação ET-N1 : Versão 4
Dezembro de 1993

Revisto em Novembro de 1995

Um Algoritmo de Geração Sequencial de Estados numa Rede Sujeita a Avarias

Teresa Martínez Gomes
e
José M. F. Craveirinha

Resumo

Neste trabalho é proposto um novo algoritmo, para gerar numa rede sujeita a avarias, sequencialmente os estados mais prováveis, por ordem decrescente de probabilidade, dada a probabilidade de avaria dos componentes.

O algoritmo desenvolvido é comparado, no que diz respeito à sua eficiência e requisitos de memória com os propostos por Lam & Li [11] e Yang & Kubat [18]. Será feita uma referência à sua aplicação a um estudo de fiabilidade/grau de serviço em redes metropolitanas digitais de comutação por circuitos com encaminhamento alternativo. Neste contexto, mostra-se que o algoritmo proposto é significativamente mais eficiente que as outras aproximações que poderiam ser aplicadas neste tipo particular de problemas.

1 Introdução

A fiabilidade nas redes de telecomunicações põe problemas muito difíceis e complexos em termos de definição, modelação e cálculo.

Tradicionalmente, até ao fim da década de 70 (ver p. ex. Meyer, 72 e Ball, 79) a análise de fiabilidade em redes de comunicação (sejam redes de telecomunicações sejam redes apenas de computadores) era expressa em termos de medidas de conectividade. Isto traduzia-se em medidas tais como a probabilidade de todos ou uma fracção dos nós de comutação de uma rede permanecerem ligados (na presença de avarias em componentes da rede). Teoricamente os problemas de análise de conectividade ou de síntese de topologias com certos requisitos de conectividade estão bem tratados embora conduzam normalmente a algoritmos de elevada complexidade (problemas 'NP hard' - isto é não polinomiais).

Na realidade, contudo, este tipo de abordagem revelou-se bastante limitado na sua perspectiva, pelas razões seguintes:

(a) os componentes das redes de comunicação são de elevada fiabilidade pelo que a probabilidade de desconexão física de nodos é normalmente muito baixa;

(b) a tendência crescente para a utilização de esquemas de encaminhamento alternativo de mensagens em redes de comunicação tenderá igualmente a diminuir a probabilidade de desconexão funcional entre pares de nodos;

(c) a avaria de componentes traduz-se normalmente numa maior ou menor degradação dos níveis de desempenho ou grau de serviço da rede, induzidos por aumentos de intensidade de tráfego em certos pontos da rede (acarretando p. ex. aumentos inaceitáveis de congestão e/ou de atrasos na transmissão de mensagens);

Surgiu assim a necessidade de reequacionar conceptual e metodologicamente as questões de análise de fiabilidade em redes de comunicação. Esta tarefa foi levada a cabo por vários autores, desde os anos 80, de que salientaremos as contribuições de Li & Silvester [12], Kubat [9, 10] e Meyer [13, 14, 15]. Uma ideia essencial, comum a todas estas abordagens do problema, é a da combinação da análise de fiabilidade com uma medida ou medidas de desempenho de rede. Nesta perspectiva, hoje dominante, as medidas de conectividade aparecem, apenas, como um dos factores a ter em conta na análise de fiabilidade das redes de telecomunicações. Poderá ainda acrescentar-se, na nossa opinião, que as medidas de conectividade têm ainda um papel de relevo a desempenhar ao nível de síntese de topologias de redes, obedecendo a certos princípios gerais de encaminhamento de tráfego, tendo em vista satisfazer certos requisitos, pelo grafo representativo da topologia. Nesta linha de pensamento se inseriram, p. ex. os trabalhos de Boesch [2], Cattermole [3] e Craveirinha & Sumner [4].

Um problema chave associado com as várias aproximações à análise da fiabilidade é a enumeração dos estados da rede a serem analisados, o qual é particularmente crítico se o número n de componentes é elevado e o cálculo das medidas de desempenho para cada estado da rede tem um custo computacional significativo.

O estudo exaustivo de fiabilidade de uma rede de telecomunicações, com n componentes sujeitos a avaria e n significativamente elevado pode-se tornar facilmente proibitivo computacionalmente, dado que o número total de estados da rede é 2^n. Surge assim a necessidade de utilizar apenas um subconjunto destes estados. Li & Silvester [12] sugerem que tendo como base esses estados, seja efectuado o cálculo das medidas de desempenho, e indicam mesmo limites inferiores e superiores para essas mesmas medidas.

A obtenção à priori de um número m adequado (que garanta uma dada probabilidade de cobertura do espaço de estados) não é trivial, pois depende fortemente das diferentes probabilidades de inoperacionalidade dos componentes presentes na rede. Numa tentativa de obter um valor "pessimista" aproximado Li & Silvester propõem utilizar um m dado pelo número de estados que seria necessário considerar se todos os componentes tivessem probabilidades de inoperacionalidade igual ao componente menos fiável da rede. Isto pode conduzir a uma estimativa para m exageradamente elevada e mesmo incomportável, se esse componente for de facto pouco fiável e se houver um número significativo de componentes comparativamente mais fiáveis.

Li & Silvester propuseram igualmente um algoritmo que permite gerar os m estados mais prováveis de uma rede. Este algoritmo não permite, caso o utilizador verifique que a

probabilidade de cobertura desejada não foi atingida. gerar de forma eficiente os estados $m+1, m+2, \ldots$ seguintes mais prováveis. Para obter mais estados seria necessário uma nova execução de todo o algoritmo. Por outro lado, o posterior cálculo das medidas de desempenho, obriga a que estes sejam armazenados na totalidade até ao momento em que são utilizados; quando a fiabilidade de alguns dos elementos sujeitos a avaria é baixo, torna-se necessário enunerar estados com múltiplas avarias, em número elevado.

Lam & Li [11], resolvem os problemas atrás enunciados ao apresentarem um novo algoritmo ORDER-II, que permite a enumeração dos estados por ordem decrescente de probabilidade de forma sequencial.

Yang & Kubat [18] propõem um novo procedimento, baseado no algoritmo de enumeração dos estados numa rede com componentes multimodo, descrito em [17], onde o problema de enumeração – sequencialmente e por ordem decrescente de probabilidade – dos estados mais prováveis de uma rede é transformado num problema de pesquisa em árvores. Em cada iteração são actualizados os intervalos inferior e superior para as medidas de desempenho, considerando que as mesmas obedecem a uma propriedade de coerência.

No presente trabalho apresenta-se um algoritmo, alternativo ao de Lam & Li que gera de forma mais eficiente estados da rede por ordem decrescente da sua probabilidade de ocorrência e sobretudo de forma mais económica no que se refere aos requisitos de memória. O algoritmo proposto é comparado com a aproximação de Yang e Kubat [18].

Este relatório encontra-se organizado como seguidamente se indica: na secção 2 são introduzidas as definições e propriedades necessárias à apresentação do algoritmo a qual é efectuada na secção 3; seguidamente na secção 4 o algoritmo é comparado com outras duas aproximações; é ainda descrita a sua a aplicação a um caso estudo na secção 5 e finalmente, na secção 6 apresentam-se algumas conclusões.

2 Conceitos Utilizados

2.1 Conceitos Básicos

Seja dada um rede com n componentes. Os componentes da rede podem estar em dois estados: o estado operacional e o estado inoperacional. A rede tem portanto 2^n estados possíveis.

Supõem-se que os componentes da rede sofrem falhas, independentemente uns dos outros, e que a probabilidade de operacionalidade do componente i é $op(i)$, com $i = 1, \ldots, n$, donde a probabilidade de inoperacionalidade é $ip(i) = 1 - op(i)$.

Em situações reais $1/2 \leq op(i) < 1$, no entanto, para que o modelo que se desenvolve em seguida seja o mais geral possível, considera-se a seguinte definição [12]:

Definição 2.1 (Ligado) *Um componente está ligado se estiver no seu estado mais provável. Caso contrário diz-se que se encontra no estado desligado. A probabilidade de um componente estar ligado designa-se por $p(i)$ e a probabilidade de estar desligado por $q(i) = 1 - p(i)$.*

Se $op(i) \geq ip(i)$ então $p(i) = op(i)$. Se $op(i) < ip(i)$ então $p(i) = ip(i)$.

Assim a probabilidade $p(i)$ traduz a probabilidade de um componente estar no seu estado mais provável.

Os componentes da rede são etiquetados de forma que:

$$1 > R(1) \geq R(2) \geq \ldots \geq R(n) > 0 \tag{1}$$

em que $R(i) = q(i)/p(i)$.

Sejam os estados da rede S_k com $k = 1, 2, \ldots, 2^n$. A sua probabilidade é dada por [12]:

$$P(S_k) = \prod_{i=1}^{n} p(i)^{1-T_i(S_k)} q(i)^{T_i(S_k)} \tag{2}$$

em que

$$T_i(S_k) = \begin{cases} 0 & \text{se } i \text{ está ligado no estado } S_k \\ 1 & \text{se } i \text{ está desligado no estado } S_k \end{cases}$$

Cada estado S_k é completamente definido pelo conjunto (desordenado) de componentes desligados que o compõem.

Sejam os S_k tal que $P(S_1) \geq P(S_2) \geq \ldots \geq P(S_{2^n})$. Então o estado mais provável (todos os componentes ligados) é o estado $S_1 = \{\}$ cuja probabilidade é:

$$P(S_1) = \prod_{i=1}^{n} p(i)$$

Não são considerados os elementos com $p(i) = 1$ $(q(i) = 0)$ porque caso existam não afectam o cálculo da probabilidade dos estados de probabilidade não nula.

Considere-se, a título de exemplo, uma rede com n componentes dos quais d estão permanentemente inoperacionais e u permanentemente operacionais. Então pode-se eliminar *do cálculo* o conjunto L destes $(d + u)$ componentes permanentemente desligados (cf. definição 2.1) pois $p(i) = 1$, $\forall i \in L$, e não afectam o cálculo dos $P(S_k)$, com $P(S_k) \neq 0$, dado por (2).

2.2 Definições e Propriedades Auxiliares

Definição 2.2 *Seja $E^{(w)}$ uma família de conjuntos cujos elementos são números inteiros entre 1 e n e estritamente ordenados por ordem crescente tal que a cardinalidade de $E^{(w)}$ é w, para $w = 1, \ldots, n$. Ou seja:*

$$E^{(w)} = \{e_1, e_2, \ldots, e_w\} : e_1 < e_2 < \ldots < e_w \wedge e_1, e_2, \ldots, e_w \in \{1, 2, \ldots, n\}$$

e $E^{(0)} = \{\}$

Existem exactamente $\binom{n}{w}$ (combinações de n, w a w) conjuntos $E_j^{(w)}$ todos diferentes, $j = 1, \ldots, \binom{n}{w}$.

Propriedade 2.1 *Cada estado S_k com exactamente w componentes desligados, designado por $S_k(w)$, pode ser representado por um e um só conjunto ordenado $E_j^{(w)}$. Pode por isso escrever-se:*

$$P(S_k(w)) = \underbrace{\left(\prod_{i=1}^{n} p(i)\right)}_{P(S_1)} \prod_{i \in E_j^{(w)}} R(i) \tag{3}$$

$$P(S_k(0) = S_1) = P(E^{(0)}) = \prod_{i=1}^{n} p_i \tag{4}$$

E inversamente, a cada conjunto ordenado $E_j^{(w)}$, corresponde um e um só conjunto desordenado S_k com exactamente w componentes desligados, $S_k(w)$.

Assim, a cada um dos conjuntos $E_j^{(w)}$ é possível associar uma probabilidade, $P(E_j^{(w)})$, dada pelas equações (3) e (4). Como é óbvio existe um número igual a $\binom{n}{w}$ de estados $S_k(w)$ diferentes.

Definição 2.3 *Seja $E^{(w)} = \{e_1, e_2, \ldots, e_w\}$ um conjunto, tal como foi definido em 2.2 Seja a função $elem(E^{(w)}, v) = e_v$, a função que devolve o v-ésimo elemento de $E^{(w)}$*

Podemos agora definir o conjunto de todos os $E^{(w)}$.

Definição 2.4 *Seja $\Phi^{(w)}$ o seguinte conjunto:*

$$\Phi^{(w)} = \{E_1^{(w)}, E_2^{(w)}, \ldots, E_{\binom{n}{w}}^{(w)}\}$$

com $P(E_1^{(w)}) \geq P(E_2^{(w)}) \geq \ldots \geq P(E_{\binom{n}{w}}^{(w)})$ tal que:

$$P(E_j^{(w)}) = P(E_{j+1}^{(w)}) \Rightarrow \begin{array}{l} elem(E_j^{(w)}, i_0) < elem(E_{j+1}^{(w)}, i_0) \\ para\ i_0 = \min\{i : elem(E_j^{(w)}, i) \neq elem(E_{j+1}^{(w)}, i)\} \end{array}$$

Esta definição garante uma identificação única para os elementos do conjunto $\Phi^{(w)}$, mesmo quando as suas probabilidades são iguais.

Propriedade 2.2 *Considerem-se os estados do sistema com exactamente w componentes desligados. Entre esses estados, o estado com maior(menor) probabilidade de ocorrência é aquele que tem desligados os componentes $1, 2, \ldots, w(n-w+1, n-w+2, \ldots, n)$, ou seja $E_1^{(w)}(E_{\binom{n}{w}}^{(w)})$.*

Propriedade 2.3 *Seja $1 \leq w < n$, então:*

$$E_1^{(w)} = \{1, 2, \ldots, w-1, w\} \tag{5}$$

$$E_2^{(w)} = \{1, 2, \ldots, w-1, w+1\} \tag{6}$$

$$E_1^{(w+1)} = \{1, 2, \ldots, w, w+1\} \tag{7}$$

E pode afirmar-se que $P(E_1^{(w)}) \geq P(E_2^{(w)}) \geq P(E_1^{(w+1)})$

Definição 2.5 *A operação* $(E^{(w)}, j) \leftarrow v$ *em que* $1 \leq j \leq w$ *é definida como sendo a atribuição do valor* v, *ao* j*-ésimo elemento de* $E^{(w)}$.

Assim:

$$elem((E^{(w)}, j) \leftarrow v), j) = v, \quad 1 \leq j \leq w$$

Definição 2.6 (Estado i-sucessivo) *Para todos o* $j = 1, 2, \ldots, \binom{n}{w}$:

1. *Diz-se que um estado* $E_j^{(1)}$ *é 1-sucessivo.*
2. *Diz-se que um estado* $E_j^{(w)}$ *é* 1*-sucessivo, com* $1 \leq w \leq n$ *se e só se*

$$elem(E_j^{(w)}, w-1) + 1 \neq elem(E_j^{(w)}, w)$$

3. *Diz-se que um estado* $E_j^{(w)}$, *com* $w > 1$, *é* w*-sucessivo se e só se*

$$elem(E_j^{(w)}, w-k) + 1 = elem(E_j^{(w)}, w-k+1), \quad k = 1, 2, \ldots, w-1$$

4. *Diz-se que um estado* $E_j^{(w)}$ *é* s*-sucessivo com* $1 < s < w$ *se e só se*

$$\begin{cases} elem(E_j^{(w)}, w-k) + 1 = elem(E_j^{(w)}, w-k+1), & k = 1, 2, \ldots, s-1 \\ elem(E_j^{(w)}, w-s) + 1 \neq elem(E_j^{(w)}, w-s+1) \end{cases}$$

Propriedade 2.4 $E_1^{(w)}$ *é* w*-sucessivo,* $w = 1, 2, \ldots, n$.

Definição 2.7 (Estado terminal) *Diz-se que um estado* $E_j^{(w)}$ *é terminal se e só se:*

$$elem(E_j^{(w)}, w) = n$$

Caso contrário diz-se que o estado é não terminal.

Definição 2.8 (Estados r-sucessivo(s) seguinte(s)) *Seja* $E_j^{(w)}$ *um estado não terminal e* s*-sucessivo. O(s) estado(s)* r*-sucessivo(s) seguinte(s) é(são) obtido(s) da seguinte forma:*

Seja $E_{j_0}^{(w)} = E_j^{(w)}$, *então*

$$E_{j_r}^{(w)} = E_{j_{r-1}}^{(w)} \tag{8}$$

$$(E_{j_r}^{(w)}, w-r+1) \leftarrow elem(E_{j_{r-1}}^{(w)}, w-r+1) + 1 \tag{9}$$

para cada $r = 1, 2, \ldots, s$. *A ordem porque surgem as equações (8) e (9) é obrigatória.*

Utilizando a definição 2.8 é possível, como será mostrado na sub-secção 2.4, gerar todos os elementos de $\Phi^{(w)}$.

Propriedade 2.5 *Os estados* $E_{j_r}^{(w)}$ *(obtidos pela definição 2.8) são r-sucessivos, com* $r = 1, 2, \ldots, s$.

Estes estados satisfazem ainda a seguinte condição:

$$P(E_j^{(w)}) \geq P(E_{j_1}^{(w)}) \geq P(E_{j_2}^{(w)}) \geq \ldots \geq P(E_{j_s}^{(w)})$$

O cálculo das probabilidades é trivial:

$$P(E_{j_r}^{(w)}) = P(E_{j_{r-1}}^{(w)}) \times \frac{R\left[\overbrace{elem(E_{j_r}^{(w)}, w-r+1)}^{v+1}\right]}{R\left[\underbrace{elem(E_{j_{r-1}}^{(w)}, w-r+1)}_{v}\right]}$$

Propriedade 2.6 *Se* $E_j^{(w)}$ *e* $E_k^{(w)}$ *são estados s-sucesivos e* $elem(E_j^{(w)}, i) = elem(E_k^{(w)}, i)$, $i = 1, 2, \ldots, w-s$, *então* $P(E_j^{(w)}) \geq P(E_k^{(w)}) \Leftrightarrow (j < k)$ *se e só se* $elem(E_j^{(w)}, w-s+1) < elem(E_k^{(w)}, w-s+1)$.

Os estados $E_j^{(w)}$ *e* $E_k^{(w)}$ *pertencem ao conjunto* $\Phi^{(w)}$, *definido em 2.4.*

A proposiçao seguinte é a base do algoritmo GeraEstados:

Proposição 2.1 *Se um estado* $E_k^{(w)}$ *com* $k \neq 1$, $w > 1$, *é t-sucessivo, com* $t < w$ *e*

$$elem(E_k^{(w)}, w-t) + 2 = elem(E_k^{(w)}, w-t+1) \tag{10}$$

então $E_k^{(w)}$ *é o estado t-sucessivo seguinte de um e um só estado* $E_j^{(w)}$.

Prova: *Seja*

$$E_j^{(w)} = E_k^{(w)} \tag{11}$$

e faça-se:

$$(E_j^{(w)}, w-v+1) \leftarrow elem(E_k^{(w)}, w-v+1) - 1, \text{ para } v = t, t-1, \cdots, 1 \tag{12}$$

O estado obtido $E_j^{(w)}$ *é um estado s-sucessivo, não terminal, com* $1 \leq t < s$.

Aplicando a definição 2.8 ao estado $E_j^{(w)}$ *é óbvio que o estado t-sucessivo seguinte de* $E_j^{(w)}$ *é o estado* $E_k^{(w)}$.

Q.E.D.

Propriedade 2.7 *Se um estado* $E_k^{(w)}$ *é t-sucessivo, com* $t < w$, $w > 1$ *e*

$$elem(E_k^{(w)}, w-t) + 2 = elem(E_k^{(w)}, w-t+1) \tag{13}$$

então, de acordo com a proposição 2.1, $E_k^{(w)}$ *é o estado t-sucessivo seguinte de um estado* $E_j^{(w)}$, $(j < k)$. *O estado* $(t+1)$*-sucessivo seguinte de* $E_j^{(w)}$ *pode ser obtido utilizando* $E_k^{(w)}$ *como* $E_{j_t}^{(w)}$ *e aplicando as equações (8) e (9) com* $r = t+1$, *da definição 2.8.*

2.3 Funções Auxiliares

As funções que seguidamente se definem têm como objectivo facilitar a apresentação do algoritmo de geração dos estados.

Definição 2.9 (Função f) *Seja f a função tal que dado um estado não terminal $E_j^{(w)}$, devolve o estado 1-sucessivo seguinte de $E_j^{(w)}$, $E_{j_1}^{(w)}$, de acordo com a definição 2.8.*

Definição 2.10 (Função g) *Seja g a função tal que dado um estado t-sucessivo $E_k^{(w)}$, com $t < w$, $w > 1$ e tal que:*

$$elem(E_k^{(w)}, w-t) + 2 = elem(E_k^{(w)}, w-t+1) \tag{14}$$

devolve o estado $E_{k_*}^{(w)}$:

$$elem(E_{k_*}^{(w)}, v) = elem(E_k^{(w)}, v), \quad v = 1, 2, \cdots, w \wedge v \neq w-t \tag{15}$$

$$elem(E_{k_*}^{(w)}, w-t) = elem(E_k^{(w)}, w-t) + 1 \tag{16}$$

De acordo com a proposição 2.1 para um estado $E_k^{(w)}$, existe um estado, $E_j^{(w)} (j < k)$, cujo estado t-sucessivo seguinte, $E_{j_t}^{(w)}$, é $E_k^{(w)}$. O estado $E_{k_*}^{(w)}$, obtido utilizando a função g é o estado $(t+1)$-sucessivo seguinte, $E_{j_{t+1}}^{(w)}$ – ver a propriedade 2.7.

Com o objectivo de comparar o algoritmo ORDER-II, proposto por Lam & Li [11], com o algoritmo GeraEstados (descrito na secção seguinte), define-se ainda a função h:

Definição 2.11 (Função h) *Seja h a função tal que que dado um estado não terminal $E_j^{(w)}$, devolve o estado $E_k^{(w+1)}$ tal que:*

$$elem(E_k^{(w+1)}, v) = elem(E_j^{(w)}, v) \text{ para } v = 1, 2, \cdots, w \tag{17}$$

$$elem(E_k^{(w+1)}, w+1) = elem(E_j^{(w)}, w) + 1 \tag{18}$$

2.4 Geração dos elementos de $\Phi^{(w)}$

O elemento de maior probabilidade pertencente a $\Phi^{(w)}$ é $E_1^{(w)} = \{1, 2, \ldots, w\}$.

Dado o estado $E_1^{(w)}$ $(w < n)$ podem obter-se os respectivo(s) estado(s) r-sucessivo(s) seguinte(s) com $r = 1, 2, \ldots, w$, utilizando a definição 2.8. Aplicando aos estados (não terminais) obtidos desta forma novamente a definição de 2.8, e assim sucessivamente, podem obter-se todos os estados $E_j^{(w)}$, com $j = 2, \ldots, \binom{n}{w}$, de acordo com a proposição seguinte:

Proposição 2.2 *Dado um estado $E_j^{(w)}$ $(1 \leq s \leq w)$ $s-$sucessivo e não terminal o número total de estados que podem ser gerados aplicando repetitivamente a definição 2.8 (estados*

r−sucessivos seguintes) até que todos os estados obtidos numa dada iteração sejam todos terminais é:

$$\binom{n - elem(E_j^{(w)}, w) + s}{s} - 1 \tag{19}$$

Prova: *Vai ser utilizada indução completa sobre s.*

Caso de Base: $(s = 1)$ *Seja*

$$E_j^{(w)} = \{e_1, e_2, \cdots, e_{w-1}, e_w\} \tag{20}$$

um estado não terminal e 1−sucessivo $(1 \leq w < n)$. *Aplicando a definição 2.8 a* $E_j^{(w)}$ *o estado:*

$$E_{j_1}^{(w)} = \{e_1, e_2, \cdots, e_{w-1}, e_w + 1\} \tag{21}$$

é gerado. O seguinte estado é portanto obtido após a b−ésima utilização da definição 2.8

$$E_{j_b}^{(w)} = \{e_1, e_2, \cdots, e_{w-1}, e_w + b\} \tag{22}$$

com $b = 1, 2, \cdots, n - e_w$.

Assim o número total de estados é $n - e_w$, *ou seja:*

$$\binom{n - e_w + 1}{1} - 1 \tag{23}$$

Hipótese de indução: Se $E_j^{(w)}$ *é um estado não terminal k−sucessivo com* $k = 1, 2, \cdots, s-1$, *então o número total de estados obtidos por aplicação repetitiva da definição 2.8 é*

$$\binom{n - elem(E_j^{(w)}, w) + k}{k} - 1 \tag{24}$$

Passo Indutivo: Considere-se que o estado $E_j^{(w)} = \{e_1, e_2, \cdots, e_{w-1}, e_w\}$ *é um estado não terminal s−sucessivo. Aplicando a definição 2.8, o(s) estado(s) r−sucessivo(s) seguinte(s)* $E_{j_r}^{(w)}$, *são obtidos com* $r = 1, 2, \cdots, s$.

Da hipótese de indução, o número de estados que podem ser obtidos de cada $E_{j_r}^{(w)}$, *com* $r = 1, 2, \cdots, s - 1$ *é conhecido e dado por:*

$$\binom{n - elem(E_{j_r}^{(w)}, w) + r}{r} - 1 = \binom{n - (e_w + 1) + r}{r} - 1 \tag{25}$$

O estado s−sucessivo é:

$$E_{j_s}^{(w)} \quad com \quad elem(E_{j_s}^{(w)}, w) = e_w + 1 \tag{26}$$

Aplicando a definição 2.8 ao estado $E_j^{(w)}$, *e repetitivamente aos estados s−sucessivos assim obtidos, após a b-ésima aplicação o estado s−sucessivo resultante é:*

$$E_{\underbrace{j_{s_{s\cdots s}}}_{b}}^{(w)} = E_{j_{sb}}^{(w)} \quad com \quad elem(E_{j_{sb}}^{(w)}, w) = e_w + b \tag{27}$$

com $b = 1, 2, \cdots, n - \epsilon_w$. *Aplicando a definição 2.8 a cada estado* $s-$*sucessivo não terminal,* $E^{(w)}_{j_{s^b}}$, *com* $b = 1, 2, \cdots, n - e_w - 1$, *além de gerar o estado* $s-$*sucessivo seguinte:*

$$E^{(w)}_{j_{s^b_{r=s}}} = E^{(w)}_{j_{s^{b+1}}} \tag{28}$$

estes estados também geram os correspondentes estados $r-$*sucessivos seguintes com* $r = 1, 2, \cdots, s-1$*:*

$$E^{(w)}_{j_{s^b_r}} \quad com \quad elem(E^{(w)}_{j_{s^b_r}}, w) = elem(E^{(w)}_{j_{s^b}}, w) + 1 = \epsilon_w + b + 1 \tag{29}$$

Os estados

$$E^{(w)}_{j_{s^b_r}} \quad com \quad r = 1, 2, \cdots, s-1 \quad \text{e } b = 1, 2, \cdots, n - \epsilon_w - 1 \tag{30}$$

podem, por sua vez, gerar um número de estados dados por (hipótese de indução):

$$\binom{n - (\epsilon_w + b + 1) - r}{r} - 1 \tag{31}$$

Assim o número total de estados gerados por aplicação da definição 2.8 sobre o estado $E^{(w)}_{j_s}$, *e repetitivamente sobre todos os estados não terminais assim obtidos, incluindo* $E^{(w)}_{j_s}$, *é:*

$$\underbrace{1}_{E^{(w)}_{j_{s^{b=n-e_w}}}} + \sum_{b=1}^{n-\epsilon_w-1} \left(\underbrace{1}_{E^{(w)}_{j_{s^b}}} + \sum_{r=1}^{s-1} \left(\underbrace{1}_{E^{(w)}_{j_{s^b_r}}} + \underbrace{\binom{n - (e_w + b + 1) + r}{r} - 1}_{devido\ a\ E^{(w)}_{j_{s^b_r}}} \right) \right) \tag{32}$$

De (25) e (32) dado o estado $s-$*sucessivo e não terminal* $E^{(w)}_j$, *o número total de estados gerados por aplicação repetitiva da definição 2.8 até que todos os estados numa dada iteração sejam terminais é:*

$$\sum_{r=1}^{s-1} \binom{n - (\epsilon_w + 1) + r}{r} + 1 + \sum_{b=1}^{n-e_w-1} \left(1 + \sum_{r=1}^{s-1} \binom{n - (\epsilon_w + b + 1) + r}{r} \right) \tag{33}$$

Reescrevendo a equação (33):

$$\sum_{b=0}^{n-e_w-1} \left(1 + \sum_{r=1}^{s-1} \binom{n - (\epsilon_w + b + 1) + r}{r} \right) \tag{34}$$

sabendo que $\sum_{b=0}^{n-e_w-1} 1$ *é igual a* $n - e_w$, *fazendo* $u = n - \epsilon_w - 1$, *e trocando a ordem dos somatórios, a expressão anterior pode ser escrita:*

$$(u+1) + \sum_{r=1}^{s-1} \sum_{b=0}^{u} \binom{(u+r) - b}{r} \tag{35}$$

e utilizando a relação:

$$\sum_{i=0}^{v} \binom{c+i}{c} = \binom{c+v+1}{c+1} = \binom{c+v+1}{v} \tag{36}$$

a expressão (35) é igual a:

$$(u+1) + \sum_{r=1}^{s-1} \binom{r+u+1}{u} = \sum_{r=0}^{s-1} \binom{r+u+1}{u} = -1 + \sum_{r=-1}^{s-1} \binom{(r+1)+u}{u} \tag{37}$$

fazendo $i = r+1$ podemos escrever:

$$-1 + \sum_{i=0}^{s} \binom{u+i}{u} \tag{38}$$

Utilizando de novo a propriedade (36), com $i = r+1$, a expressão anterior pode ser escrita:

$$\binom{u+s+1}{s} - 1 \tag{39}$$

e substituindo u pelo seu valor, $n - \epsilon_w - 1$, a prova fica completa.

Q.E.D.

3 Descrição do Algoritmo

A aproximação utilizada, para proceder ao armazenamento dos estados candidatos a serem seleccionados, é semelhante à utilizada em [11], a qual utiliza uma fila com prioridades (implementada como uma "heap" de acordo com [8]). Uma "heap" é uma árvore binária completa com a propriedade de que o valor associado com cada nó não é menor (maior) de que o valor dos nós filhos, caso existam. Assim a raiz de uma "heap" tem sempre o maior (menor) valor. Numa "heap" com k elementos, um nó pode ser retirado ou inserido em operações de ordem $O(\log_2 k)$. No contexto deste texto, cada nó é um estado candidato, e o seu valor é a probabilidade desse estado.

3.1 A ideia base do Algoritmo

A ideia base do algoritmo consiste em ir seleccionando o estado mais provável entre os elementos de $\Phi^{(w)}$, $w = 1, 2, \ldots, n$, já gerados, mas ainda não seleccionados. Os estados candidatos encontram-se armazenados numa "heap" (por ordem decrescente do seu valor). O primeiro elemento de $\Phi^{(w+1)}$ só precisa estar presente na "heap" depois de terem sido seleccionados os dois primeiros elementos de $\Phi^{(w)}$ (ver propriedade 2.3).

Na implementação aqui apresentada, o primeiro elemento de $\Phi^{(w+1)}$ é adicionado à "heap" assim que o primeiro elemento de $\Phi^{(w)}$ é seleccionado – por razões que se prendem com a simplificação da exposição.

Para cada estado seleccionado, não terminal e $s-$sucessivo, S_k, tal que $\#S_k = w$, os estados $r-$sucessivo(s) seguinte(s) poderiam ser gerados de acordo com a definição 2.8 e adicionados à "heap" de estados candidatos (este procedimento garante que todos os estados pertencentes a $\Phi^{(w)}$ serão gerados). Este processo colocaria na "heap" um grande número de estados candidatos que provavelmente nunca seriam seleccionados. Uma solução alternativa, e mais eficiente, seria gerar o estado $1-$sucessivo seguinte, para cada estado seleccionado não terminal e $s-$sucessivo, tal que $\#S_k = w$ (S_k corresponde a algum estado $E_j^{(w)}$) e colocá-lo na "heap". A geração dos eventualmente restantes estados $r-$sucessivos seguintes, $E_{j_r}^{(w)}$, com $r = 2, 3, \cdots, s$ é adiada. Esses elementos serão gerados apenas quando necessário (ver propriedade 2.5), ou seja quando o estado $E_{j_1}^{(w)}$ for seleccionado da "heap"; nesse caso o estado E_{j_2} deve ser gerado de acordo com a propriedade 2.7, e adicionado à "heap", e de forma semelhante para os restantes estados E_{j_r}, para $r = 3, 4, \cdots, s$.

A utilização repetitiva da propriedade 2.7 sobre cada $E_{j_r}^{(w)}$ para cada $r = 1, 2, \ldots, s-1$, é um procedimento equivalente a utilizar a definição 2.8 sobre um estado $E_j^{(w)}$ não terminal e $s-$sucessivo.

O que atrás foi dito implica que para cada estado seleccionado, dois estados (no máximo) terão de ser adicionados à "heap": o estado $1-$sucessivo seguinte do último estado seleccionado (se esse estado é não terminal) e o estado obtido utilizando a propriedade 2.7 (caso o estado seleccionado satisfaça a proposição 2.1).

3.2 O algoritmo GeraEstados

No algoritmo GeraEstados de cada vez que um estado seleccionado, com w componentes desligados, é não terminal, o estado $1-$sucessivo seguinte é adicionado à "heap". Por outro lado se o estado seleccionado, tem w elementos desligados, o algoritmo utiliza a propriedade 2.3 para adicionar o estado $E_1^{(w+1)}$ à "heap", caso o estado seleccionado tenha sido o estado $E_1^{(w)}$.

O número de elementos desligados do estado candidato, presente na "heap", com maior número de elementos desligados, será registado numa variável auxiliar, $w_{\max}$. Se o estado seleccionado é $s-$sucessivo, com $s < w$ (e nesse caso é certamente diferente de $E_1^{(w)}$) e satisfaz a proposição 2.1 então o estado obtido utilizando a propriedade 2.7 é adicionado à "heap".

Em seguida, é formalizado o algoritmo GeraEstados. O estado mais provável é $S_1 = \emptyset$. O segundo estado mais provável é o estado $S_2 = \{1\}$.

Algoritmo GeraEstados:
Inicializa: $i := 2$; $S_2 := \{1\}$; $w_{\max} = 1$; Árvore $:= \emptyset$;
Repete
  Se (S_i é não terminal) Então adiciona $f(S_i)$ à Árvore ; FimSe ;
  Se (S_i é s-sucessivo)
    E ($\#S_i > s$)
    E ($elem(S_i, w-s) + 2 = elem(S_i, w-s+1)$)

Então adiciona $g(S_i)$ à Árvore
Senão
Se ($w_{\max} = \#S_i$) E ($\#S_i < n$)
Então
adiciona $h(S_i)$ à Árvore ;
$w_{\max} = \#S_i + 1$;
FimSe ;
FimSe ;
S_{i+1} := estado na raiz da Árvore ;
retira a raiz à Árvore ;
$i := i + 1$;
Até que tenham sido seleccionados os estado desejados.

Neste algoritmo "*adiciona $h(S_i)$ à Árvore;*" pode ser substituído por "*adiciona $E_1^{(\#S_i+1)}$ à Árvore;*".

É importante referir que as operações de adicionar um elemento à árvore e de retirar a raiz da árvore incluem os procedimentos de reordenação necessários à preservação da estrutura ordenada da "heap". Na Tabela 1, é apresentado um exemplo ilustrativo da forma como o algoritmo evolui para $n = 4$ e $\{p(i)\} = \{0.9, 0.95, 0.99, 0.995\}$, A forma da "heap" durante as primeiras seis iterações é apresentada, antes da remoção da raiz da árvore, assim como a probabilidade do estado seleccionado no fim de cada iteração.

No que se refere ao cálculo de complexidade deste algoritmo deve notar-se que adicionar um nó à árvore ou retirar a raiz da árvore custa $O(\log_2 k)$ operações, sendo k é o número de elementos na árvore. Se o estado presente na "heap", com maior número de elementos desligados, tem $w_{\max}$ elementos desligados, então o armazenamento dos elementos de um estado precisa de $O(w_{\max} + 1)$ operações. As restantes operações não interferem com o cálculo da complexidade do algoritmo. O número de operações por iteração é $O(w_{\max} + \log_2 k)$, o qual é significativamente mais baixo que o valor usual de $O(n)$.

Escreve-se $w_{\max}$ em vez de $(w_{\max} + 1)$ porque no algoritmo GeraEstados existe um e apenas um estado na "heap" pertencente ao conjunto $\Phi^{(w_{\max})}$.

Após a obtenção de m estados o número de operações é $O(mw_{\max} + m \log_2 \bar{m})$, e os requisitos de memória são $O(\bar{m} w_{\max})$, considerando que a dimensão máxima da "heap", após a selecção de m estados é $\bar{m}$. Mostrar-se-á em seguida que $\bar{m}$ é menor do que m. Após longa experimentação verificou-se que em situações realistas $\bar{m}$ é significativamente menor do que m (ver os exemplos da Tabela 2).

No algoritmo GeraEstados a dimensão da "heap" aumenta uma unidade de cada vez que o estado seleccionado, S_k, é não terminal e $s-$sucessivo, com $\#S_k > s$, $\#S_k > 1$ e satisfaz a proposição 2.1. A "heap" também aumenta de uma unidade de cada vez que o estado seleccionado é o primeiro elemento de algum conjunto $\Phi^{(w)}$, o que acontece apenas uma única vez para cada conjunto $\Phi^{(w)}$.

A fracção de estados seleccionados, pertencentes a um conjunto $\Phi^{(w)}$ que correspondem a um aumento de uma unidade na "heap", no algoritmo GeraEstados é (ver proposição A.1 no apêndice):

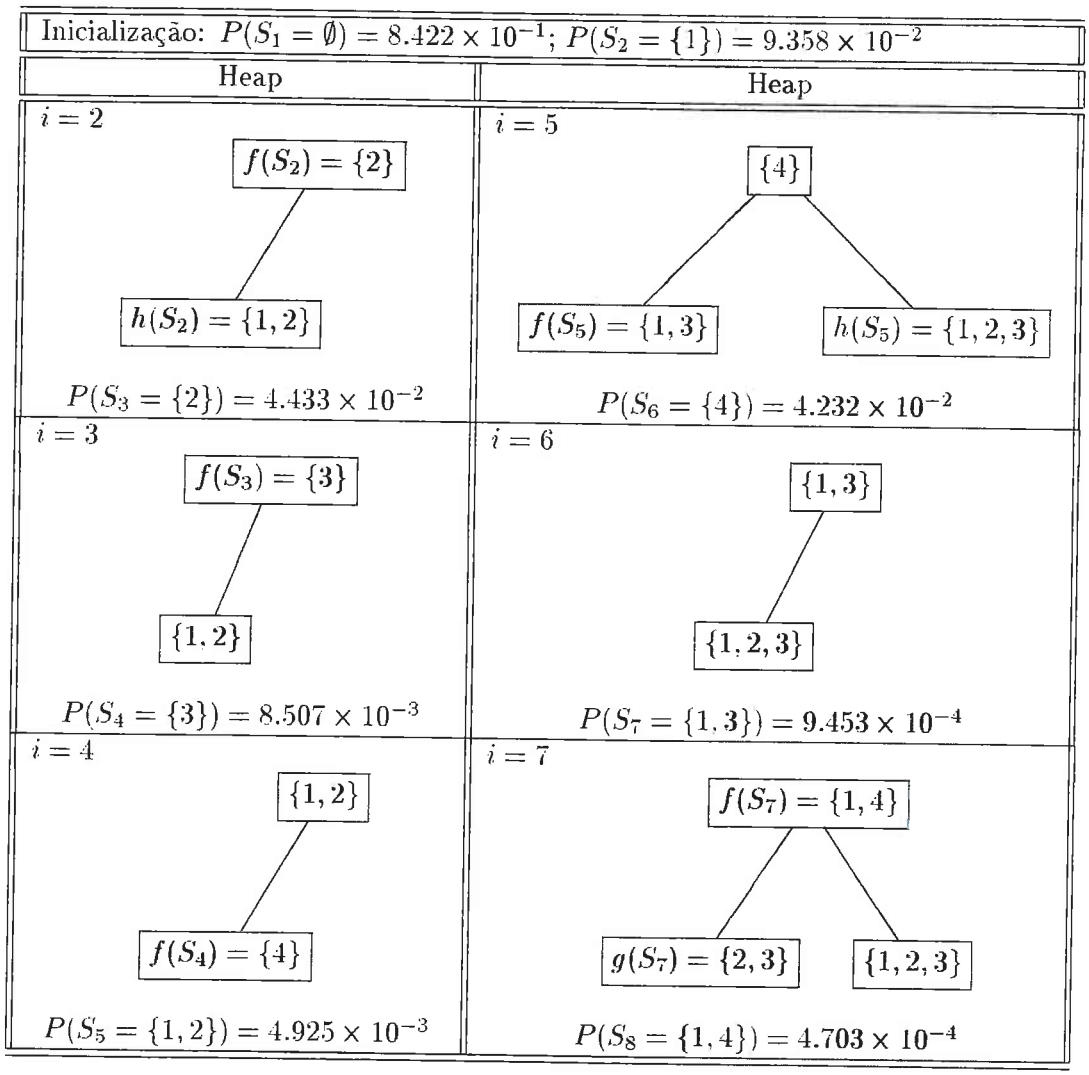

Tabela 1: Algoritmo GeraEstados: a "heap" durante as primeiras seis iterações.

$$F_{StG}(n,w) = \frac{w(n-w)}{n(n-1)} \tag{40}$$

o qual é $\ll 1$ para n grande quando comparado com w, a qual é uma relação típica na maior parte das aplicações.

Notando que o valor (40), para n fixo, aumenta com w para $w = 1, 2, \cdots, (n+1) \div 2$ e seguidamente diminui para $w = (n+1) \div 2 + 1, \cdots, n$, definindo $w_m = \min(w_{\max} - 1, (n+1) \div 2)$, um limite superior para $F_{StG}(n,w)$, durante a execução do algoritmo GeraEstados ocorre para $w = w_m$, para um dado n.

Seja $\hat{m} = 2mF_{StG}(n, w_m)$ uma estimativa para a dimensão máxima da "heap". Através de experimentação intensiva, $\hat{m}$ provou ser, em situações realistas, um bom limite superior para a dimensão máxima da "heap" durante a execução do algoritmo GeraEstados. O factor 2 foi introduzido para levar em conta que os estados não são seleccionados de

forma aleatória entre os elementos de cada $\Phi^{(w)}$, especialmente quando $\binom{n}{w} \gg m$. Este factor foi obtido empiricamente, após numerosas experiências. Tendo com objectivo o dimensionamento das estruturas de dados necessárias à execução do algoritmo, é interessante possuir um estimativa para $\bar{m}$ antes da execução do algoritmo, considerando que m é o número máximo de estados a seleccionar.

Um limite superior para $\bar{m}$, que normalmente está muito longe (por excesso) do valor real, pode ser dado, utilizando a propriedade A.1, por:

$$\dot{m} = \sum_{i=0}^{w_{\max}-2} \binom{n-2}{i}, \quad w_{\max} < n \tag{41}$$

em que se pressupõe que o valor do lado direito é inferior a m (caso contrário este limite não tem qualquer interesse prático).

4 Comparando com Outras Aproximações

4.1 Comparando com o algoritmo ORDER-II

Tendo como objectivo a comparação da eficiência entre o algoritmo proposto neste texto e o algoritmo ORDER-II em [11], segue-se este último, reescrito na presente notação:

Algoritmo ORDER-II:
Inicializa: $i := 2$; $S_2 := \{1\}$; Árvore := $\emptyset$;
Repete
  Se S_i é não-terminal
    Então
      adiciona $f(S_i)$ à Árvore ;
      adiciona $h(S_i)$ à Árvore ;
  FimSe ;
  S_{i+1} := estado na raiz da Árvore ;
  retira a raiz à Árvore ;
  $i := i + 1$;
Até que tenham sido seleccionados os estados desejados.

Em [11] este algoritmo é apresentado com tendo complexidade $O(nm + m \log_2 m)$, e requisitos de memória $O(nm)$ se for utilizado para gerar m estados.

Nesta implementação, o armazenamento dos elementos de um novo estado é proporcional ao número de elementos desligados adicionado de uma unidade. Assim se o estado com maior número de elementos desligados na "heap" tem $w_{\max}$ elementos (e normalmente haverá muito mais do que um estado com esse número de elementos desligados) então a complexidade é $O((w_{\max}+1)m + m \log_2 m)$ e os requisitos de memória são $O((w_{\max}+1)m)$.

No algoritmo ORDER-II, de cada vez que um estado é seleccionado (com w elementos desligados) é não terminal, dois estados são adicionados à "heap": o estado 1−sucessivo

seguinte desse estado e o estado com $(w+1)$ elementos desligados, obtido utilizando a função h.

Embora a complexidade seja semelhante para ambos os algoritmos (com vantagem para GeraEstados), pode mostrar-se que o algoritmo GeraEstados é significativamente mais eficiente, especialmente no que concerne aos requisitos de memória.

A principal diferença entre os dois algoritmos reside no facto de que o segundo estado adicionado à "heap", pelo algoritmo GeraEstados tem probabilidade igual ou superior ao segundo estado adicionado no algoritmo ORDER-II – facto este que irá conduzir a uma "heap" com menor número de elementos durante a execução de GeraEstados quando comparado com ORDER-II.

Seja o último estado seleccionado S_i, o qual é representado por $E_j^{(w)}$, com probabilidade P_i. Seja $E_j^{(w)}$ um estado não terminal e s-sucessivo, com $s < w$. O primeiro estado adicionado à "heap", $E_{j_1}^{(w)}$, é comum a ambos os algoritmos. No algoritmo ORDER-II, o segundo estado $h(S_i)$ tem probabilidade:

$$P_{\text{Ord}} = P_i R(u+1), \text{ onde } u = elem(E_j^{(w)}, w) \tag{42}$$

e considerando que $E_j^{(w)}$ satisfaz a proposição 2.1, no algoritmo GeraEstados o segundo estado $g(S_i)$ tem probabilidade:

$$P_{\text{StG}} = P_i \frac{R(v+1)}{R(v)}, \text{ com } v = elem(E_j^{(w)}, w-s) \wedge u = v+s+1 \, com \, 1 \leq s < w \tag{43}$$

e $v+1 < u+1 \Longrightarrow R(v+1) \geq R(u+1)$. Portanto:

$$\frac{R(v+1)}{R(v)} > R(u+1) \Longrightarrow P_{\text{StG}} > P_{\text{Ord}} \tag{44}$$

tendo presente que $1 > R(1) \geq R(2) \geq \cdots \geq R(n) > 0$, é correcto concluir que, na maior parte dos casos $P_{\text{StG}} \gg P_{\text{Ord}}$. Como exemplo numérico considere-se $R = [0.25, 0.22, 0.19, 0.02, 0.01]$ e $S_6 = \{1,3\}$, o qual é 1-sucessivo, não terminal; $u = 3$, $s = 1$, $v = 1$, e $P(S_6) = 2.54 \times 10^{-2}$, então $P(g(S_6) = \{2,3\}) = 2.24 \times 10^{-2} \gg P(h(S_6) = \{1,3,4\}) = 5.08 \times 10^{-4}$.

As probabilidades do segundo estado adicionado à "heap" só são iguais, em ambos os algoritmos, quando esse estado é o primeiro elemento de um conjunto $\Phi^{(w)}$, e esta situação ocorre apenas um número de vezes dado por $w_{\max} - 1$.

Em ORDER-II, quando um estado não terminal é seleccionado, dois estados são adicionados à "heap" antes da raiz ser retirada; isto implica que para cada estado não terminal seleccionado a dimensão da "heap" aumenta de uma unidade. A fracção de estados seleccionados num conjunto $\Phi^{(w)}$, que correspondem a um aumento de uma unidade na "heap", para o algoritmo ORDER-II (ver proposição B.1), é:

$$F_{\text{Ord}}(n, w) = \frac{n-w}{n} \tag{45}$$

n	Algoritmo	$w_{\max}$	$\bar{m}$	$\hat{m}$	Ordem dos Requisitos de Memória	Ordem da Complexidade /10^6
20	ORDER-II	11	111865	———	1.34×10^6	14.39
	GeraEstados	11	2598	263158	2.86×10^4	11.17
50	ORDER-II	6	452365	———	3.17×10^6	12.89
	GeraEstados	6	11017	91837	6.61×10^4	9.71
100	ORDER-II	5	489329	———	2.94×10^6	12.45
	GeraEstados	5	10839	38788	5.42×10^4	9.20
200	ORDER-II	4	497155	———	2.48×10^6	11.96
	GeraEstados	4	4597	14850	1.84×10^4	8.08
500	ORDER-II	4	498999	———	2.49×10^6	11.96
	GeraEstados	4	4716	5975	1.89×10^4	8.10
1000	ORDER-II	3	498075	———	1.99×10^6	11.46
	GeraEstados	3	493	1998	1.48×10^3	5.97

Tabela 2: Comparando os algoritmos, para $m = 500000$.

o qual é próximo de 1 quando n é grande comparado com w, e esta é a relação mais comum entre n e w.

No entanto, um limite superior para a dimensão da "heap" pode ser obtido utilizando a propriedade B.1:

$$\hat{m} = \min\left(m, \sum_{i=1}^{w_{\max}-1}\left[\binom{n}{i} - \binom{n-1}{i-1}\right]\right), \quad w_{\max} < n \tag{46}$$

No algoritmo ORDER-II o número de nós na "heap", depois de seleccionar m estados ($m \ll 2^n$), é próximo de m, mas no algoritmo GeraEstados é muito menor.

O algoritmo GeraEstados é mais rápido (tipicamente duas a três vezes) e utiliza menos memória do que o algoritmo ORDER-II porque, para obter o mesmo número de estados m, gera um menor número de estados – de probabilidade igual ou superior – normalmente com um menor número de elementos desligados, e armazena-os numa "heap" com menor número de elementos.

Na Tabela 2 apresentam-se o valor de $w_{\max}$, o valor do número de nós na "heap", $\bar{m}$, o valor da estimativa para $\bar{m}$ dada por $\hat{m}$ (apenas para GeraEstados), os requisitos de memória e as ordens de complexidade para ORDER-II e GeraEstados respectivamente.

Os valores apresentados na tabela pressupõe que ambos os algoritmos param após gerarem meio milhão de estados, para vários valores de n e probabilidade dos componentes estarem ligados $p(i)$, $i = 1, 2, \cdots, n$ tal que $n = 20 <> p(i) \in [0.9, 0.99995]$, $n = 50, 100, 200 <> p(i) \in [0.99, 0.99999]$ e $n = 500, 100 <> p(i) \in [0.999, 0.999999]$. No cálculo dos valores nas colunas correspondentes à ordem dos requisitos de memória e à ordem de complexidade, foi utilizado o número máximo de elementos na "heap" durante a execução de cada um dos algoritmos.

O caso $n = 20$ e $m = 500000$ é um caso especial, no sentido em que quase metade de todos os estados possíveis foram já seleccionados, pelo que $\bar{m}$, para ORDER-II, é muito menor do que m mas ainda assim muito maior do que $\bar{m}$ do algoritmo GeraEstados. Uma breve análise, para $n = 50, 100, 200, 500, 1000$, mostra que a dimensão da "heap", $\bar{m}$, para o algoritmo GeraEstados é muito pequena quando comparada com m. No caso do algoritmo ORDER-II o valor de $\bar{m}$ é sempre próximo de m, como aliás já tinha sido previsto. A estimativa $\hat{m}$ é maior do que $\bar{m}$ em todos os casos, e o factor de 2 só é efectivamente necessário para $n = 500$, devido ao facto de que um número significativo de estados foi seleccionado de $\Phi^{(w_{\max}-1)}$ e $\binom{n}{w_{\max}-1} \gg m$.

No que diz respeito ao limite superior $\dot{m}$ (46) para $\bar{m}$, (o qual não é apresentado na Tabela 2), o mesmo é próximo de $\bar{m}$ for ORDER-II (na maior parte dos casos), mas $\dot{m}$ (41) sobre-estima largamente o valor de $\bar{m}$ para o GeraEstados, com a excepção de $n = 1000$.

4.2 Comparando com a aproximação de Yang e Kubat

Yang & Kubat [18] propõem um novo procedimento baseado no algoritmo de enumeração de estados numa rede com componentes multimodo (apresentado em [17]) onde o problema de enumeração dos estados mais prováveis de uma rede é transformado num problema de de pesquisa em árvores.

Sucintamente, Yang & Kubat [18] definem uma árvore binária especial G, de altura n, tal que cada vértice no nível l $(0 \leq l \leq n-1)$ tem exactamente dois filhos e cada vértice no nível n é uma folha. (em que n é o número de elementos sujeitos a avaria na rede). O endereço de um vértice u no nível l é definido como sendo um vector de dimensão l $(x_1, x_2, \cdots, x_l)$ tal que $x_i = 1(0)$ se e só se é o único caminho, por exemplo, $a_0(\text{raiz}), a_1, \cdots, a_{l-1}, a_l(u)$ da raiz até u, a_i é o filho esquerdo (direito) de a_{i-1}. Existe uma correspondência de um para um entre o endereço de uma folha v em G e o vector $\mathbf{x}$ de um estado da rede. O problema de enumeração dos estados mais prováveis é equivalente à identificação dos endereços das folhas mais pesadas de G.

Yang & Kubat [18] utilizam o algoritmo para obter limites superior e inferior para a desempenhabilidade da rede, após cada iteração, considerando que a medida de desempenho satisfaz uma propriedade de coerência. Uma medida de desempenho é dita possuir a propriedade de coerência se os elementos operacionais num estado A são um sub-conjunto dos elementos operacionais num estado B, então o desempenho do estado A é menor ou igual ao de B. Esta propriedade é típica das medidas de desempenho baseadas na conectividade.

O número de operações por iteração é $O(n)$, e embora a estrutura da árvore binária só seja efectivamente criada para as as folhas identificadas e para para os seus antecessores, tal poderá representar uma grande quantidade de dados, para n e m elevados.

Por conseguinte, GeraEstados tem complexidade inferior por iteração e menores requisitos de memória do que este algoritmo [18]. No entanto o algoritmo de Yang & Kubat, para medidas de desempenho que possuem a propriedade de coerência, fornece limites superior e inferior mais próximos, para o mesmo número de iterações.

Por outro lado muitas aplicações em redes de telecomunicações inter-centrais, com

encaminhamento alternativo, tal como acontece no caso estudo em que o algoritmo GeraEstados foi utilizado, a propriedade de coerência não é necessariamente observada por algumas das medidas de desempenho mais significativas. Neste tipo de redes o efeito mais frequente das falhas é o eclodir de sobre-cargas na rede nas áreas mais afectadas. Em redes de comutação por circuitos, os efeitos do encaminhamento alternativo na região de sobre-carga são geralmente referidos como o problema de estabilidade [16], o qual tem sido estudado de forma extensiva, nomeadamente no contexto das redes telefónicas (ver p. ex. [7]). Este problema consiste num aumento anormal das probabilidades de bloqueio ponto a ponto em certas condições de sobre-carga implicando que o tráfego médio transportado pela rede pode diminuir nessas condições e seguidamente aumentar devido ao efeito de falhas adicionais. Esta situação resulta do facto de falhas adicionais impedirem certos fluxos de tráfego de acederem às áreas da rede mais congestionadas e desse modo permitirem o aumento do tráfego transportado associado a outros fluxos de tráfego, de médias mais elevadas. Note-se que uma das estratégias utilizadas, para prevenir estes efeitos, é o corte, de acordo com certas regras, do acesso de certos fluxos de tráfego (normalmente tráfego de transbordo) à áreas mais congestionadas da rede, o que é equivalente a falhas adicionais *fictícias* na rede original. Assim sendo, o tráfego médio transportado na rede, ou a probabilidade de bloqueio médio da rede, que são medidas de desempenho fundamentais neste tipo de rede de comunicações, poderão não satisfazer a propriedade de coerência que é a base da obtenção de melhores limites inferior e superior para as medidas de desempenho no algoritmo [18].

5 Aplicação a um Caso Estudo Real

Um modelo computacional para a análise da fiabilidade/desempenho de uma rede inter-centrais de comutação por circuitos, de grandes dimensões, foi implementado em colaboração com a operadora de telecomunicações portuguesa [5]. O objectivo fundamental do modelo é a avaliação integrada da fiabilidade – desempenho de uma rede inter-centrais de grande dimensão, especificada em termos dos seus componentes físicos (incluindo os elementos técnicos que implementam as funções de comutação e transmissão) e a rede funcional associada, descrita pelos seus centros de comutação, tráfego oferecido ponto a ponto, feixes funcionais e algoritmo de encaminhamento.

O modelo utilizado, além de levar em conta os problemas específicos destas redes, nomeadamente a sua dimensão, utiliza uma avaliação multi-paramétrica de desempenho da rede, que inclui várias medidas de grau de serviço. A dimensão do problema tratado, ficará mais clara com a seguinte informação: a rede funcional no nosso caso estudo possui 46 nós (centrais de comutação) e 693 feixes funcionais; a rede física é representada por 757 elementos, de diferente natureza, dos quais 28 foram considerados 100% fiáveis. Cada elemento utilizado na descrição da rede funcional representa a agregação de componentes técnicos específicos. Esta agregação teve ser efectuada de forma a tornar o estudo computacionalmente viável.

Os dados disponíveis referem-se à fiabilidade dos componentes da rede física, não à fiabilidade dos elementos da rede funcional, e embora em teoria seja possível calcular a

probabilidade de falha deste últimos com base na probabilidade de falha dos primeiros, essa tarefa é extremamente difícil e demorada, quando as redes são de grande dimensão e/ou complexidade, tal como acontece no nosso caso. Tal implicou que os estados de falha fossem definidos na rede física, mas os seus efeitos, em termos de medidas de desempenho, tivessem de ser calculados nos estados correspondentes da rede funcional, a qual requer uma representação separada.

Finalmente, os componentes da rede sujeitos a avaria são de grande fiabilidade, e do ponto de vista do teletráfego os efeitos dominantes dos estados de falha mais prováveis são sobre-cargas em algumas partes da rede, aumentos na congestão e chamadas e nos tempos médios de espera, ou noutras palavras, uma degradação da qualidade de serviço entre pares de centrais, e perda do rendimento associado, por parte do operador, no que concerne ao tráfego telefónico, de dados, ou de qualquer outro tipo de tráfego que seja transportado pela rede. Uma rede com estas características pode ser considerada um sistema que se degrada graciosamente (*gracefully degrading system*) no sentido definido em [15].

O algoritmo GeraEstados foi utilizado para seleccionar eficientemente os estados mais prováveis da rede física, por ordem decrescente de probabilidade, de uma forma tipicamente interactiva. O processo de geração dos estados pode ser interrompido, temporariamente ou definitivamente interrompido se qualquer das seguintes condições for verificada:

- foi atingida a cobertura probabilística desejada do espaços de estados;
- foi atingido um número máximo m de estados, considerado viável, dado o esforço computacional máximo envolvido no cálculo das medidas de desempenho (estimado com base no custo de obtenção de todas as medidas de desempenho para um estado);
- o último estado seleccionado tem uma probabilidade inferior a um valor mínimo, especificado como limite inferior ainda significativo.

Os valores iniciais para qualquer um dos factores atrás mencionados poderão ser inicializados no inicio da execução do algoritmo, e quando qualquer um deles é satisfeito, o analista poderá decidir, com base na sua experiência, e nas características da rede em estudo, decidir acerca da paragem da execução do programa, ou se o factor em causa deverá ser alterado de forma a permitir a continuação da geração dos estados. Esta aproximação permitirá ultrapassar situações conflituosas entre esses objectivos, recorrendo à intervenção e validação do analista. Convém ter presente que o cálculo das medidas de desempenho desejadas, baseadas no cálculo das probabilidades de bloqueio ponto a ponto por cada estado seleccionado, tem um custo computacional significativo. Chama-se ainda a atenção para o facto de que neste tipo de redes algumas das medidas de desempenho chave (tal como a probabilidade de bloqueio médio) não satisfazem necessariamente a propriedade de coerência, tal como foi definida em [18].

Tendo presente a caracterização do problema em estudo e a estrutura conceptual em que foi abordado, foi utilizada a seguinte metodologia para a análise da fiabilidade, cujas características dominantes são:

- uma selecção interactiva, através do algoritmo GeraEstados dos estados mais prováveis da rede de componentes.

- A utilização de um modelo de tráfego bi-paramétrico, para descrever todos os fluxos de tráfego ponto a ponto, em todos os feixes da rede funcional, incluindo uma descrição completa dos fluxos de tráfego em cada feixe (médias, variâncias e probabilidades de bloqueio marginais têm de ser calculadas); este modelo é a base do cálculo dos bloqueios ponto a ponto, através de algoritmos numéricos iterativos [6].

- Definição e cálculo de um conjunto multidimensional de parâmetros de desempenho, de vários tipos, pesados pelas probabilidades dos estados correspondentes. Estas medidas desempenho referem-se ao desempenho da rede, tanto ao nível da comunicação entre centrais como ao nível de desempenho global da rede (valores médios para todos os pares de centrais) como ainda ao nível de desempenho de feixe. Ou seja no nosso modelo, medidas de desempenhabilidade são definidas simultaneamente através de valores médios e através da probabilidade de exceder certos níveis de desempenho. É também efectuada uma avaliação do efeito económico das falhas, calculado através do custo associado com o incremento de tráfego perdido, com base no custo do Erlang.hora, para cada fluxo de tráfego.

O modelo implementado provou ser uma ferramenta flexível para a avaliação do efeito das falhas dos elementos da rede física sobre o desempenho da rede global. É também um instrumento que permite avaliar e comparar soluções alternativas para a rede, resultando, por exemplo, de alterações nas capacidades dos feixes, na disposição dos sistemas de transmissão ou ainda nos esquemas de encaminhamento alternativo utilizados.

6 Conclusões

Foi apresentado um novo algoritmo para gerar, de forma eficiente e flexível, os estados mais prováveis numa rede sujeita a avarias, o qual não requer a satisfação da propriedade de coerência nas medidas de desempenho a avaliar. Esta aproximação reduz a complexidade por iteração e consequentemente o algoritmo proposto tem complexidade inferior aos propostos em [11, 18]. Os requisitos de memória são também bastante mais baixos do que os necessitados por ORDER-II ou pelo algoritmo em [18] se n e m são grandes.

A aproximação de Yang & Kubat [18] conduz a melhorias significativas na convergência dos limites inferior e superior das medidas de desempenho, quando comparado com o algoritmo proposto (uma vez que na maior parte dos casos utilizará um menor número de iterações) mas apenas quando as medidas de desempenho de interesse satisfazem a propriedade de coerência. Este facto limita de alguma forma a sua aplicação a certas redes de telecomunicações inter-centrais com encaminhamento alternativo, onde alguns dos parâmetros de desempenho chave não satisfazem necessariamente a propriedade de coerência.

Foi ainda efectuada uma referência a uma aplicação bem sucedida do algoritmo GeraEstados, no contexto de um estudo de fiabilidade–grau de serviço numa rede urbana digital de comutação por circuitos, com encaminhamento alternativo.

A Algoritmo GeraEstados

A proposição e propriedade seguintes referem-se ao comportamento do algoritmo GeraEstados, para $n > 1$.

Proposição A.1 *A fracção de estados seleccionados, no conjunto $\Phi^{(w)}$, que correspondem a um aumento de uma unidade na "heap", para o algoritmo GeraEstados é:*

$$\frac{w(n-w)}{n(n-1)} \tag{47}$$

Prova: *Seja $E^{(w)} = \{e_1, e_2, \cdots, e_w\}$ um estado s-sucessivo ($s < w$, $w > 1$) que satisfaz a proposição 2.1:*

$$e_{w-i} = u - i, \quad for \quad i = 0, 1, 2, \cdots, s-1 \quad and \quad u = w+1, w+2, \cdots, n \tag{48}$$

$$e_{w-s} = u - (s+1); \tag{49}$$

Para cada valor de u, os restantes elementos de $E^{(w)}$, $e_1, e_2, \cdots, e_{w-(s+1)}$, definem estados, diferentes entre si, que satisfazem a proposição 2.1: sabendo que, por definição de $E^{(w)}$, o maior valor que $e_{w-(s+1)}$ pode tomar é $u-(s+2)$, esses estados são em número:

$$\binom{u-(s+2)}{w-(s+1)} \quad com \quad u = w+1, w+2, \cdots, n \tag{50}$$

Então, pela equação (50), o número de estados s-sucessivo ($s < w$, $w > 1$) que satisfazem a proposição 2.1, (utilizando (36)) é:

$$\sum_{u=w+1}^{n} \binom{u-(s+2)}{w-(s+1)} = \sum_{i=0}^{n-w-1} \binom{w-(s+1)+i}{w-(s+1)} \tag{51}$$

$$= \binom{n-(s+1)}{w-s} \tag{52}$$

$$= \underbrace{\binom{n-(s+2)}{w-s}}_{\text{não terminal}} + \underbrace{\binom{n-(s+2)}{w-(s+1)}}_{\text{terminal}} \tag{53}$$

No algoritmo GeraEstados o número de elementos na "heap" aumenta de uma unidade sempre que o estado seleccionado é não terminal e satisfaz a proposição 2.1. O número de estados nessas condições pertencentes ao conjunto $\Phi^{(w)}$ é dado pela soma de todos os estados não terminais na equação (53), para $s = 1, 2, \cdots, w-1$:

$$\sum_{s=1}^{w-1} \binom{n-(s+2)}{w-s} \tag{54}$$

A "heap" aumenta também de uma unidade sempre que o estado seleccionado é o primeiro elemento de algum conjunto $\Phi^{(w)}$, o que ocorre apenas uma vez para cada conjunto $\Phi^{(w)}$, com $w < n$. Assim o número total de estados, no conjunto $\Phi^{(w)}$, que correspondem a um aumento de uma unidade na "heap", para o algoritmo GeraEstados é:

$$N_{s*} = 1 + \sum_{s=1}^{w-1} \binom{n-(s+2)}{w-s} \tag{55}$$

$$= \sum_{i=0}^{w-1} \binom{(n-w-2)+i}{i} \tag{56}$$

Fazendo $c = n - w - 2$ e utilizando a equação (36):

$$N_{s*} = \binom{w+c}{c+1} \tag{57}$$

Finalmente, substituindo c pelo seu valor:

$$N_{s*} = \binom{n-2}{n-w-1} \tag{58}$$

Para $w = 1$ a "heap" apenas aumenta de uma unidade quando o estado seleccionado é $E_1^{(w=1)}$; então a expressão (58) também é válida para $w = 1$.

A fracção de estado seleccionados, no conjunto $\Phi^{(w)}$, que correspondem a um aumento de uma unidade na "heap", é:

$$\frac{N_{s*}}{\binom{n}{w}} = \frac{w(n-w)}{n(n-1)} \tag{59}$$

Q.E.D.

Propriedade A.1 *O número total de estados seleccionados que correspondem a um aumento de uma unidade na "heap" para o algoritmo GeraEstados, considerando que entre os elementos de $\Phi^{(i)}$, para $i = 1, 2, \cdots, w$, todos os estados nessas condições foram seleccionados, é, utilizando (58):*

$$\sum_{i=1}^{w} \binom{n-2}{n-i-1} = \sum_{i=0}^{w-1} \binom{n-2}{i} \tag{60}$$

B Algoritmo ORDER-II

A proposição e propriedades seguintes, referem-se ao comportamento do algoritmo ORDER-II, para $n > 1$.

Proposição B.1 *A fracção de estados seleccionados, no conjunto $\Phi^{(w)}$, que corresponde, a um aumento de uma unidade na "heap", para o algoritmo ORDER-II, é:*

$$\frac{n-w}{n} \tag{61}$$

Prova:

O número total de estados terminais, $N_t(w)$, num conjunto $\Phi^{(w)}$ é:

$$N_t(w) = \binom{n-1}{w-1} \tag{62}$$

A fracção de estados seleccionados, no conjunto $\Phi^{(w)}$, que correspondem a um aumento de uma unidade na "heap", para o algoritmo ORDER-II é:

$$1 - \frac{N_t(w)}{\binom{n}{w}} = \frac{n-w}{n} \tag{63}$$

Q.E.D.

Propriedade B.1 *O número total de estados seleccionados, no algoritmo ORDER-II, que correspondem a um aumento de uma unidade da "heap", considerando que entre os elementos de $\Phi^{(i)}$ para $i = 1, 2, \cdots, w$, todos os estados não terminais foram seleccionados, é:*

$$\sum_{i=1}^{w} \left[\binom{n}{i} - \binom{n-1}{i-1} \right] \tag{64}$$

Referências Bibliográficas

[1] M. O. Ball, Computing Network Reliability, Operations Research 27 (1979) 821-838.

[2] F. T. Boesch, "On Graphs of Invulnerable Communication Nets", *IEEE Trans. on Circuit Theory*, Vol. CT-17, No. 2, May (1970).

[3] K. W. Cattermole and J. P. Sumner, "Communication Networks based on the Product Graph", *Proc. IEE*, Vol. 124, No. 1, January (1977).

[4] J. Craveirinha and J. P. Sumner, "Application of the Product Graph to a Large Multiexchange Digital Network", *Proc. IEE*, Part I, Vol. 136, No. 3, (1989) 189-196.

[5] J. Craveirinha, T. Gomes e J. Esteves, "A Model for Reliability Analysis of a Large Multiechange Network", in Proc. of the ITC Regional International Teletraffic Seminar: "Teletraffic Seminar for Evolving Networks", Pretoria, South Africa, September (1995) 184-191.

[6] J. S. Esteves and J. Craveirinha, An Efficient Algorithm for Computing Node-to-Node Blocking Probabilities in Teletraffic Networks with Arbitrary Routing, in Proc. of the 6th International Symposium on Applied Stochastic Models and Data Analysis, Crete, ed. Jansen and Skiadas, Vol. 1, (World Scientific, 1993) 231-242.

[7] O. Hashida, S. Nakajima, K. Okada and M. Shinohara, Congestion Mechanisms and Overlead Control in Telephone Networks, in Proc. of the 9th International Teletraffic Congress, Torremolinos, Spain (1979).

[8] E. Horowitz and S. Sahni, Fundamental of Data Structures (Pitman, 1976).

[9] P. Kubat, Assessing the Throughput and Reliability in Communication and Computers Networks, IEEE Trans. on Reliability 37 (1988) 308-311.

[10] P. Kubat, Estimation of Reliability for Communication/Computer Networks – Simulation/Analytic Approach, IEEE Transactions on Communications 37 (1989) 927-933.

[11] Y. F. Lam and V. O. Li, An Improved Algorithm for Performance Analysis of Networks with Unreliable Components, IEEE Transactions on Communications 34 (1986) 496-497.

[12] V. O. K. Li and J. A. Silvester, Performance Analysis of Networks with Unreliable Components, IEEE Transactions on Communications 32 (1984) 1105-1110.

[13] J. F. Meyer, On Evaluating the Performability of Degradable Computing Systems, IEEE Trans. on Computers 29 (1980) 720-731.

[14] J. F. Meyer, "Performance Evaluation Techniques and Tools", *Proceedings of the 5ºth ITC Seminar on ISDN Traffic Issues*, Lago Como, Itália, (1987).

[15] J. F. Meyer, Performability: a retrospective and some pointers to the future, Performance Evaluation 14 (1992) 139-156.

[16] T.-K. G. Yum and M. Schwartz, Comparison of Routing Procedures for Circuit-Switched Traffic in Nonhierarchical Networks, IEEE Transactions on Communications 35 (1987) 535-544.

[17] C.-L. Yang and P. Kubat, Efficient Computation of Most Probable States for Communication Networks with Multimode Components, IEEE Transactions on Communications 37 (1989) 535-538.

[18] C.-L. Yang and P. Kubat, An Algorithm for Network Reliability Bounds, ORSA Journal on Computing 2 (1990) 336-345.