GOVERNO DO ESTADO DO RIO DE JANEIRO
SECRETARIA DE ESTADO DE CIÊNCIA E TECNOLOGIA
FUNDAÇÃO DE APOIO À ESCOLA TÉCNICA
CENTRO DE EDUCAÇÃO PROFISSIONAL EM TECNOLOGIA DA INFORMAÇÃO
FACULDADE DE EDUCAÇÃO TECNOLÓGICA DO ESTADO DO RIO DE JANEIRO -
FAETERJ/PETRÓPOLIS

Aplicação da Evolução Diferencial na Formação de Ataque da Equipe SIR2D de Simulação de Futebol 2D

Bernardo Barbosa de Araujo

Petrópolis - RJ

Outubro, 2016

Bernardo Barbosa de Araujo

Aplicação da Evolução Diferencial na Formação de Ataque da Equipe SIR2D de Simulação de Futebol 2D

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Tecnólogo em Tecnologia da Informação e da Comunicação da Faculdade de Educação Tecnológica do Estado do Rio de Janeiro - FAETERJ/Petrópolis, como requisito parcial para obtenção do título de Tecnólogo em Tecnologia da Informação e da Comunicação.

Orientador:
D.Sc. Eduardo Krempser da Silva

Petrópolis - RJ

Outubro, 2016

Folha de Aprovação

Trabalho de Conclusão de Curso sob o título *"Aplicação da Evolução Diferencial na Formação de Ataque da Equipe SIR2D de Simulação de Futebol 2D"*, defendida por Bernardo Barbosa de Araujo e aprovada em 27 de Outubro de 2016, em Petrópolis - RJ, pela banca examinadora constituída pelos professores:

Prof. D.Sc. Eduardo Krempser da Silva
Orientador
Faculdade de Educação Tecnológica do Estado
do Rio de Janeiro - FAETERJ/Petrópolis

Prof. M.Sc. Thiago Tavares Magalhães
Laboratório Nacional de Computação
Científica - LNCC

Prof. M.Sc. Alberto Torres Angonese
Faculdade de Educação Tecnológica do Estado
do Rio de Janeiro - FAETERJ/Petrópolis

Declaração de Autor

Declaro, para fins de pesquisa acadêmica, didática e tecnico-científica, que o presente Trabalho de Conclusão de Curso pode ser parcial ou totalmente utilizado desde que se faça referência à fonte e aos autores.

Bernardo Barbosa de Araujo

Petrópolis, em 27 de Outubro de 2016

Agradecimentos

Aos meus pais Oscar Filho e Márcia Nunes pelo apoio aos meus estudos, mesmo de longe durante os últimos anos. As minhas irmãs e familiares que de alguma forma durante esse período também me apoiaram.

Ao meu orientador de Iniciação Científica, professor Marcos Todorov, pelas orientações aos meus trabalhos no Laboratório Nacional de Computação Científica - LNCC, que resultaram na produção desta monografia.

Aos professores Eduardo Krempser e Alberto Angonese pelas orientações aos meus trabalhos no Laboratório de Sistemas Inteligêntes e Robótica - SIRLab.

Aos amigos do LNCC e do SIRLab, em especial a: Caio Graciani, Johnathan Fercher e Lucas Borsatto, pelo apoio e ajuda durante os meus estudos em sistemas e controle.

A todos os amigos, professores e funcionários da FAETERJ-Petrópolis e do LNCC que de alguma forma deram contribuições a minha formação.

Epígrafe

"A natureza é um templo augusto, singular,
Que a gente ouve exprimir em língua misteriosa;
Um bosque simbolista onde a árvore frondosa
Vê passar os mortais, e segue-os com o olhar."

(Charles Baudelaire)

Resumo

Desde o início das pesquisas no desenvolvimento de ferramentas de busca em 1994, os algoritmos responsáveis pelo ranqueamento de um conjunto de documentos interligados têm sido a parte mais desafiadora na implementação desses sistemas. Dentre esses algoritmos, o *PageRank* do Google obteve grande destaque no final dos anos 90, fazendo com que cada vez mais o buscador fosse uma ferramenta indispensável na navegação pela *Web*. Neste trabalho são apresentados os resultados obtidos das simulações dos modelos responsáveis pelo cálculo do *PageRank*.

Lista de Figuras

Sumário

1 Introdução

A crescente quantidade de informação na *Web* nos últimos anos, fez das ferramentas de busca algo indispensável na coleta de informação. O algoritmo *PageRank* é a técnica chave do Google, proposta inicialmente em [3] no ano de 1998. O Google é um dos mais bem sucedidos buscadores e o *PageRank* é com certeza um dos fatores deste sucesso.

É possível encontrar na literatura artigos recentes [14, 13] que ilustram o crescente interesse da comunidade de sistemas e controle no tema. Tal interesse advem das diversas dificuldades que são encontradas no cálculo do *PageRank*, devido a sua complexidade. O problema é de grande dimensão e em geral trata-se de um sistema mal comportado. Além disso, a variabilidade da *Web* torna necessária a atualização frequente do cálculo. Ademais, a possibilidade do cálculo ser efetuado através de recursos computacionais distribuídos [11] evidencia o caráter desafiador deste problema.

Para o cálculo do *PageRank* algumas considerações devem ser feitas e um fator crítico a ser levado em conta é o tamanho da estrutura da *Web*. A *Web* até 2010 era composta por pouco mais de 8 bilhões de páginas [12], lembrando ainda que esse número está em constante crescimento. Atualmente a estimação do *Ranking* das páginas é feita de forma centralizada no Google, onde toda a coleta de dados da *Web* é feita por rastreadores, ou *Crawlers*, que a acessam de forma autônoma.

A estrutura da *Web* pode ser descrita como um grafo $\mathcal{G} = (\nu, \varepsilon)$ [10], sendo cada página tratada como um vértice $\nu = \{1, 2, ..., n\}$ e cada *link* como uma aresta dada por $\varepsilon \subseteq \nu \times \nu$. Assim, se uma página i possui um *link* para j, então $(i, j) \in \varepsilon$. O algoritmo do *PageRank* também está associado a conceitos da área de sistemas e controle. Seu cálculo dá-se por uma equação de diferença [4], cuja a matriz de transição é do tipo estocástica, devido a possibilidade do conjunto de páginas da *Internet* poderem ser modeladas como uma cadeia de Markov de estados discreto [5].

Este trabalho trata-se de uma revisão dos modelos propostos para o algoritmo do *PageRank* em [13]. Assim, inicialmente foi necessário um estudo dos conceitos por trás dos modelos

matemáticos do algoritmo, para que em seguida pudessem ser realizadas as simulações e serem feitas conclusões a partir delas. Foram realizadas simulações desde os modelos mais simples a aqueles que estão associados a situações mais mal comportadas. Por fim, são analisados os resultados e comparados os modelos e recursos usados nas simulações.

Assim, serão abordados conceitos de sistemas estocásticos no Capítulo II. Um histórico e detalhes sobre o *PageRank* são apresentados no Capítulo III. Os modelos que regem o algoritmo são abordados no Capítulo IV. Os resultados das simulações dos modelos estão relatados no Capítulo V. Por fim, o trabalho é concluído e são propostas futuras pesquisas e implementações no tema.

2 *Sistemas Estocásticos*

Existe atualmente uma vasta literatura dedicada ao estudo de sistemas estocásticos, devido à crescente complexidade das aplicações modernas [9, 7]. O estudo dessa classe de sistemas foi impulsionado em grande parte pelos avanços em telecomunicações e em computação distribuída. Entretanto, os processos estocásticos também estão presentes em diversas áreas da ciência como, por exemplo, economia e biologia [2].

Neste segundo capítulo serão abordados alguns conceitos de sistemas estocásticos, conceitos esses que estão por trás dos modelos apesentados no trabalho. Sendo assim, serão discutidos conceitos relacionados às equações de diferença, matrizes estocásticas e cadeias de Markov [6, 8].

2.1 A Equação de Diferença

A simulação do *PageRank* é dada por uma equação de diferença com vetores e matrizes estocásticas. Cada um dos valores do vetor representa a importância de uma página. A simulação consiste em calcular o próximo vetor de estados, de forma recursiva, até que se atinja um valor estacionário. Assim, o cálculo do *PageRank* é representado pela seguinte equação de diferença de primeira ordem:

$$x(k+1) = Ax(k), \qquad k = 0, 1, 2, \ldots. \tag{2.1}$$

2.2 Definição e Propriedades de Matrizes Estocásticas

Os sistemas estocásticos possuem uma matriz de transição específica, conhecida como matriz estocástica, matriz probabilidade ou ainda matriz de Markov. A matriz estocástica é usada para descrever a transição de uma cadeia de Markov. Cada uma de suas entradas é um número real não negativo representando uma probabilidade de transição. Essas matrizes podem assumir

três formatos: o de coluna estocástica, o de linha estocástica e o duplamente estocástica.

Uma matriz coluna estocástica tem como propriedade ter cada coluna com soma igual a um, matriz A em (4.2),

$$A = \begin{pmatrix} \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{2} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & 1 & 0 \\ 0 & \frac{1}{3} & 0 & 0 \end{pmatrix}, \tag{2.2}$$

e uma matriz linha estocástica tem como propriedade ter cada linha com soma igual a um, matriz B em (2.3),

$$B = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}. \tag{2.3}$$

Já uma matriz duplamente estocástica tem a propriedade de cada coluna e linha ter soma igual a um, matriz C em (2.4),

$$C = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}. \tag{2.4}$$

E por fim, o vetor estocástico, ou vetor probabilidade, possui cada um de seus elementos como números reais não negativos e com soma igual a um, vetor v em (2.5),

$$v = \begin{pmatrix} 0 \\ \frac{1}{2} \\ 0 \\ \frac{1}{2} \end{pmatrix}, \tag{2.5}$$

seguindo o mesmo que para uma matriz de única coluna ou linha.

2.3 Cadeias de Markov

Os sistemas sujeitos a saltos markovianos pertencem a classe de sistemas estocásticos. Podem se apresentar em tempo contínuo ou discreto. E de forma análoga, podem possuir um espaço de estados contínuos, ou discretos. Neste trabalho, considera-se sistemas lineares cuja transição, ou salto, de uma configuração para outra é representada por uma cadeia de Markov. A cadeia de Markov é um processo estocástico cuja estimação do estado futuro só depende do estado presente. Com isso, dizemos que tal processo de salto não possui memória a respeito de seus estados anteriores.

Assim, considere $\theta = \theta(k), k = 0, 1, ...$ uma cadeia de Markov no espaço de estados discreto $n = 1, 2, \ldots, W$. Com a seguinte propriedade:

$$P\big(\theta(k+1) = j \mid \theta(k) = i_k, \theta(k-1) = i_{k-1}, \ldots, \theta(0) = i_0\big) = P\big(\theta(k+1) = j \mid \theta(k) = i_k\big). \quad (2.6)$$

denominada propriedade de Markov. Observa-se na propriedade (2.6), que no contexto do *PageRank i* trata-se da página onde encontra-se um determinado navegador e j é a página para a qual ele está seguindo. Vale lembrar que o lado direito da igualdade da expressão (2.6) aparece simplificado pelo fato dos estados anteriores serem irrelevantes para este processo estocástico.

Uma cadeia de Markov com espaço de estados discreto pode ser interpretada como um grafo. A figura 2.1 mostra um exemplo desta abordagem. Neste trabalho considera-se que partindo-se de um determinado estado, os estados seguintes a serem alcançados são identicamente prováveis. Portanto, dado que de cada um dos nós 1,2,3 e 4 é possível ir para qualquer outros dois, a probabilidade de ir para cada um dos outros dois seguintes é de 50%.

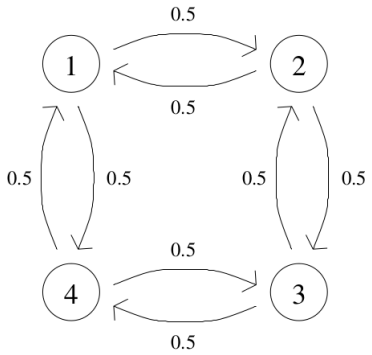

Figura 2.1: Ilustração de uma simples cadeia de Markov.

Sendo assim, a matriz estocástica referente ao grafo seria a matriz (2.7),

$$D = \begin{pmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \end{pmatrix}. \tag{2.7}$$

E a simulação em questão seria então dada pela equação (2.8),

$$x(k+1) = Dx(k), \qquad k = 0,1,2,\ldots. \tag{2.8}$$

3 O Algoritmo PageRank

O *PageRank* é o principal algoritmo por trás das engrenagens de busca do *Google*, criado em 1998 por Lawrence Page e Sergey Brin 3.1 para atribuir pesos a páginas da *Web*. Ele tem como função atribuir um valor numérico para cada elemento em um conjunto de documentos, em que na *Internet* tratariam-se de páginas *Web*. Tendo como principal objetivo a identificação das páginas mais importantes aos usuários da rede, de forma a atribuir valores maiores as páginas mais importantes.

Figura 3.1: Lawrence Page e Sergey Brin criadores do *PageRank* e fundadores do Google.

O *Google* descreve o *PageRank* da seguinte forma: *"O PageRank reflete nossa percepção de importância das páginas da Web de forma a considerar mais de 500 milhões de variáveis e 2 bilhões de termos. Páginas que são consideradas importantes recebem um maior valor de PageRank, sendo assim, possuem maior possibilidade de aparecerem no topo de um resultado de busca."*

A respeito da atribuição de importância às páginas, a página mais importante será a mais visitada e a página mais visitada será aquela que está associada a um maior número de *links*. De forma mais precisa o número de visitas de uma página vai estar associado aos *links* de saída que ela possui. Contudo, uma página também pode ser considerada importante por estar próxima a uma página de alto *PageRank*. Algumas páginas famosas e seus *PageRanks* são mostrados na tabela 3.1.

Nome da Página	PageRank
google.com	10
ebay.com	9
espn.com	8
ge.com	7
generalmills.com	6

Tabela 3.1: Valores atribuídos a algumas páginas da *Web* pelo Google em 2007.

No trabalho em que foi proposto o *PageRank* em 1998 [3], é resolvida a questão de como construir um sistema de larga escala para explorar informações adicionais da *Web* de forma a lidar com coleções de páginas e *links* não controlados, onde qualquer usuário da rede pode publicar ou retirar uma página a cada instante.

Entretanto, o *PageRank* é só uma pequena parte do sistema de buscas do Google. O sistema do buscador parte de uma *query* que é feita na barra de pesquisa do Google 3.2, em seguida as palavras que estão contidas nesta *query* são procuradas no conjunto de páginas indexadas pelo buscador. Por fim, o *PageRank* faz o ranqueamento das páginas que possuem a palavra pesquisada.

Figura 3.2: Barra de pesquisa do Google.

Contudo, apesar de toda a complexidade do sistema de busca, o maior problema por trás do buscador é fazer o ranqueamento desse número massivo de páginas. Não por acaso os fundadores escolheram o nome Google para representar a ferramenta de busca que haviam criado, uma vez que a palavra vem de googol, ou 10^{100}, para representar a massividade da *Web*.

3.1 As Ferramentas de Busca

O histórico de sucesso das ferramentas de busca está associado ao número de páginas indexadas pelos buscadores. Quanto mais páginas indexadas, melhores eram os resultados de busca pois estavam mais adequados ao que procura o navegador. Desde 1994 tecnologias de busca têm tido que alcançar o crescimento da Web. Como pode-se constatar na tabela 3.2, uma das primeiras ferramentas de busca, a *World Wide Web Worm - WWWW* [15] possuia 110 mil páginas *Web* indexadas. No entanto, em 1997 as melhores ferramentas de busca precisavam ter um índice com cerca de 100 milhões de páginas e em 1998 o Google surge com 518 milhões de páginas indexadas.

Ano	Buscador	Nº de Páginas Indexadas
1994	WWWW	110 mil
1997	AltaVista	100 milhões
1998	Google	518 milhões
2016	Google	4.8 bilhões

Tabela 3.2: Número de páginas indexadas pelos buscadores.

3.2 O Problema do *PageRank*

O cálculo do *PageRank* está sujeito a um grande problema, o custo computacional. Através dos dados apresentados na seção anterior, em função da grande dimensão da *Web* pode-se constatar que esse cálculo não é nada trivial. Além do custo computacional, outros problemas surgem ao considerarmos os casos em que um navegador aleatório [1] não segue a estrutura de *Hyperlink*. Assim, são três os principais problemas da simulação do cálculo do *PageRank*: o problema dos *Dark Holes* ou Buracos Negros, o problema da navegação descontínua por páginas não diretamente ligadas e o problema do cálculo do *PageRank* de um conjunto massivo de documentos.

O *PageRank* pode ser pensado como um modelo que simula o comportamento de um usuário que age de forma aleatória. Considerar que existe um navegador que vai acessando páginas aleatoriamente, de forma a clicar somente em *links* na página em que se encontra, consiste no modelo mais simples de cálculo do *PageRank*.

Outra importante situação que precisa ser levada em consideração é a possibilidade de um usuário acessar uma página sem links de saída, como por exemplo, um link que o direcione para

um documento PDF, como ilustrado na figura 3.3. Neste caso o usuário poderia usar o botão de voltar do *Browser* para sair deste tipo de buraco negro na rede, o que simplifica possíveis problemas no cálculo do *PageRank*.

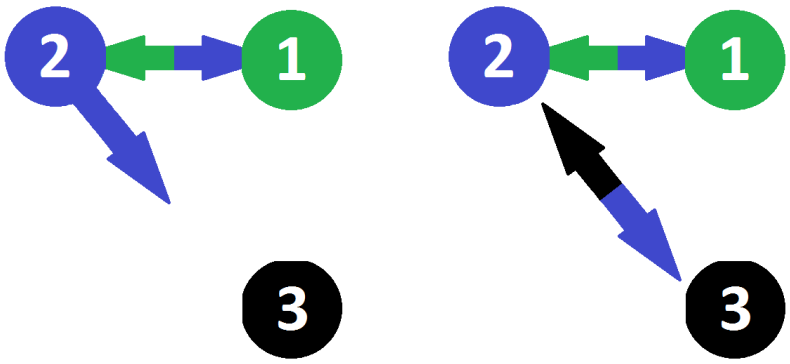

Figura 3.3: O buraco negro da *Web*.

Um outro problema surge ao considerar-se que o usuário pode saltar de uma página para outra sem seguir a estrutura de *links*, por ficar entediado, ou por um outro motivo qualquer. Um exemplo de como isso poderia ocorrer, é o caso do usuário estar na página do Laboratório Nacional de Computação Científica - LNCC e seguindo a estrutura de *Hyperlink*, clicar em um *link* que o leve à página do governo federal. No entanto, o usuário de repente altera o *Uniform Resource Locator - URL* e vai para um *site* de notícias que não está diretamente ligado aos outros dois *sites*. Este exemplo é ilustrado na figura 3.4.

Figura 3.4: Navegação entre páginas indiretamente conectadas.

Este problema foi apontado pela primeira vez no artigo que deu origem ao Google em 1998 [3], e foi recentemente apelidado em [13] de *Teleportation Model*.

A massividade da *Web* é uma das principais questões por trás do cálculo do *PageRank*, considerando que a dimensão da *Web* pode tornar o ranqueamento de páginas impossível. No entanto, a solução para este problema é o uso de algoritmos distribuídos, sobre os quais a cada ano novas publicações são feitas [14]. Neste contexto, a fim de melhorar a performance do processamento, o cálculo a partir do modelo distribuído é realizado com a ajuda de vários conputadores num *cluster*, como o da figura 3.5. Sendo assim, os recursos computacionais já necessários para abrigar a *Web* também seriam usados para o cálculo de *PageRank*.

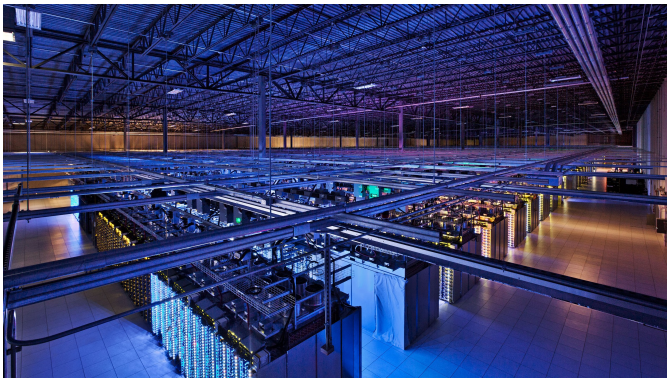

Figura 3.5: Cluster do Google.

4 *Definição dos Modelos*

O cálculo do *PageRank* possui desde um modelo mais simples, até outras adaptações deste modelo devido aos problemas descritos na seção anterior. A estimação do *PageRank* é feita por uma equação de diferença, a qual possui uma matriz de transição do tipo estocástica. Na primeira seção deste capítulo é demostrada a construção dessa matriz. Na sequência são apresentados o *Power Method*, modelo mais simples do cálculo do *PageRank*, o *Teleportation Model*, modelo que trata da navegação descontínua e o modelo por trás dos algoritmos distribuídos.

4.1 A Matriz *Hyperlink*

A construção da matriz *hyperlink* é feita com base na estrutura da *Web*, ou ainda do grafo que a representa. Assim, considerando que o grafo da figura 4.1 representa um conjunto de páginas interligadas, seus elementos são definidos da seguinte forma:

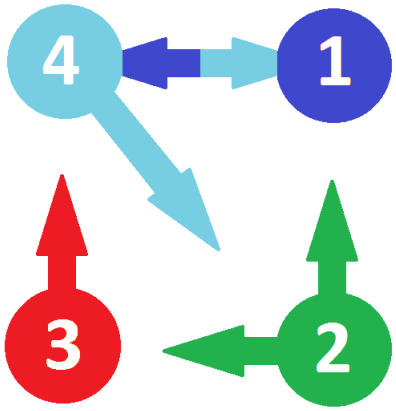

Figura 4.1: Grafo representando *links* entre páginas *Web*.

- n: como os nós, que representam as páginas,

- $\mathcal{E}$: como as arestas, que representam os *links* entre as páginas.

Para representar o *link* entre duas páginas específicas usa-se i e j, de forma que o i esteja associado à página de saída e j com a de entrada. Considerando então o caso de um vértice i estar conectado a um j, tem-se $(i, j) \in \mathcal{E}$.

A matriz de transição que é construída a partir do grafo tem a seguinte propriedade para cada um de seus elementos:

$$a_{ij} = \begin{cases} \dfrac{1}{n_i}, & \text{caso } (i,j) \in \mathcal{E}, \\ 0, & \text{caso contrário.} \end{cases} \tag{4.1}$$

Cada elemento a_{ij} é formado a partir das relações entre os nós do grafo. Para o nó 1 por exemplo, constata-se que ele só possui relações com o nó 4, por tanto, para um navegador que encontra-se em 1 tem-se 100%, de certeza de que seu próximo passo será para 4. Já no caso do nó 2 pode-se observar que o navegador pode ir para 1 ou 3, ou seja, ele tem 50% de chance de ir para cada um desses dois nós.

$$A = \begin{pmatrix} 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 1/2 \\ 0 & 1/2 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix} \tag{4.2}$$

E assim constroi-se a matriz de transição (4.2) de forma que todas as suas colunas respeitem a propriedade:

$$\sum_{i=1}^{n} x_i = 1. \tag{4.3}$$

Vale ressaltar que partindo de qualquer nó, todos os outros nós ligados a este possuem a mesma probabilidade de serem acessados. Ou seja, em uma página um *link* ou outro não terão maior probabilidade de serem acessados por chamarem mais atenção, ou por qualquer outro motivo.

Sendo assim, o grafo, ou o conjunto de páginas, trata-se de uma cadeia de Markov com espaço de estados de dimensão n. Onde a estimação de uma posição seguinte apenas depende da atual.

4.2 O *Power Method*

Este é o modelo mais simples para a estimação do *PageRank*. Esta abordagem constitui-se de uma equação de diferença, a qual possui uma matriz de transição coluna estocástica e um vetor estocástico. O vetor estocástico é formado pelos valores das variáveis aleatórias da cadeia de Markov, ou seja os valores das páginas. O valor de *PageRank* de um conjunto de n páginas da *Web* é definido como o vetor $x \in \mathbb{R}^{n\times 1}$ que satisfaz as seguintes equações:

$$x = Ax, \quad x \geq 0, \quad \sum_{i=1}^{n} x_i = 1, \tag{4.4}$$

onde $A \in \mathbb{R}^{n\times n}$ é a transposta da matriz de transição da cadeia de Markov.

Assim, o método usual para obtenção do *PageRank* é o chamado *Power Method* [13], que alcança o *PageRank* através da seguinte iteração:

$$x(k+1) = Ax(k), \quad k \geq 0, \quad \text{com} \quad x(0) = x_0, \tag{4.5}$$

onde $x_0 \in \mathbb{R}^{n\times 1}$ é uma condição inicial positiva de soma igual a um.

4.2.1 Questões de Convergência do *Power Method*

Após um número considerável de iterações e em um número considerável de casos, o *Power Method* atinge a distribuição limite com os valores de *PageRank* de cada página. Além disso, independente da condição inicial, ao final das iterações os valores de *PageRank* tendem a ser os mesmos.

Isso ocorre devido a matriz de transição ser irredutível, o que reflete no navegador aleatório sempre conseguir chegar a uma certa página partindo de uma outra qualquer, mesmo que essas duas páginas não estejam diretamente conectadas. Observando o grafo da figura 4.1, pode-se observar que não existe um *link* direto entre 1 e 3, para chegar-se em 3 a partir de 1 o usuário teria de ir de 1 para 4, de 4 para 2 e de 2 para 3.

No caso de uma página apenas possuir *links* de entrada, como por exemplo, ao acessar-se um documento PDF, o problema é contornado ao admitir-se que o usuário poderia usar o botão de volta do *Browser*, criando um *link* de saída da página. Assim, a matriz mantem-se irredutível, garantindo que para a maioria das condições iniciais os valores de *PageRank* sejam os mesmos ao atingir-se a distribuição limite. No entanto, existem alguns casos em que não há garantia de

convergência, e por tanto o modelo do *Power Method* tem de sofrer algumas alterações.

4.3 O *Teleportation Model*

O valor de *PageRank* é estimado após inúmeras iterações entre o vetor de valores das páginas e a matriz de transição estocástica. No entanto mesmo observando os valores das páginas em um grande horizonte de tempo, existem casos em que eles não convergem. O modelo do *Power Method* é atraente pela sua simplicidade, entretanto alguns problemas fazem com que ele não ofereça uma garantia de convergência. O maior desses problemas é devido a possibilidade do usuário fazer uma navegação descontínua no grafo, em que ele saltaria de uma página para outra, de forma a não seguir a estrutura de *links* da rede. Todavia, existe um modelo para resolver o problema da navegação com saltos, modelo este conhecido como *Teleportation Model*.

O *Teleportation Model* é uma estratégia reconhecida para que, através de uma pequena modificação na matriz A, o método convirja globalmente para o PageRank. Essa modificação na matriz A é representada como uma combinação convexa de duas matrizes, em que a matriz de transição agora passa a ser M, dada por:

$$M = (1-m)A + \frac{m}{n}\mathbf{1}\mathbf{1}^T, \tag{4.6}$$

onde $M \in \mathbb{R}^{n\times n}$, $m \in (0,1)$ e $\mathbf{1} \in \mathbb{R}^{N\times 1}$. Em que $\mathbf{1}\mathbf{1}^T$ é uma matriz preenchida de 1's, resultante do produto entre um vetor de 1's $\mathbf{1}$ e um vetor de 1's transposto $\mathbf{1}^T$. Vale lembrar também que n é o número de nós do grafo, ou número de páginas, e que A continua sendo a mesma matriz de transição, apenas é substituida por M na equação apresentada no *Power Method*.

4.4 O Modelo dos Algoritmos Distribuídos

Por fim, um último problema é considerado antes de serem feitas as simulações, a massividade da *Web*. O fato da *internet* ter um grande número de páginas, faz com que o tempo necessário para o cálculo e ranqueamento das páginas seja extremamente grande. A fim de tornar o cálculo do *PageRank* menos custoso, e melhor explorar os recursos computacionais dos servidores disponíveis na web, uma alternativa é o emprego de algoritmos distribuídos. A agregação de tais experimentos aleatórios faz então com que o *power method* se torne um sistema linear com saltos Markovianos do seguinte tipo:

$$x(k+1) = A_{\theta(k)}x(k), \quad k \geq 0, \quad \text{com} \quad x(0) = x_0. \tag{4.7}$$

Em que a matriz de transição A agora passa a ser várias matrizes *links* distribuídas $A_i \in \mathbb{R}^{n\times n}$, $i = 1,2,...,n$. Sendo cada uma das matrizes distribuídas definidas da seguinte forma:

- a i-ésima coluna de A_i coincide com a i-ésima coluna de A,
- a j-ésima entrada diagonal de A_i é igual a um,
- para $j = 1,...,n$, $j \neq i$, todas as outras entradas a_{ij} são iguais a zero.

A ideia agora é de que cada página fará o seu cálculo de *PageRank*, por isso são criadas as matrizes distribuídas. Com relação a construção das matrizes, pode-se observar que elas seguem a seguinte ideia. As colunas das matrizes são formadas pelas relações de saída de cada um dos nós, ao mesmo tempo que cada matriz *link* distribuída está associada a um nó do grafo da figura 4.1. Assim a coluna 1 de A (4.2) é copiada para a primeira coluna de A_1 (4.8), o restante da diagonal pricipal é preenchida com 1's e o restante dos elementos é zero. O mesmo se repete para as outras matrizes distribuídas.

$$A_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix} \quad A_2 = \begin{pmatrix} 1 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1/2 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.8}$$

$$A_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \quad A_4 = \begin{pmatrix} 1 & 0 & 0 & 1/2 \\ 0 & 1 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.9}$$

Acoplando agora o modelo das matrizes distribuídas ao modelo do *Teleportation* tem-se (4.10),

$$x(k+1) = (1-\hat{m})A_{\theta(k)}x(k) + \frac{\hat{m}}{N}\mathbf{1}\mathbf{1}^T, \quad k \geq 0, \quad \text{com} \quad x(0) = x_0, \tag{4.10}$$

onde $\hat{m}$ é definido como:

$$\hat{m} = \frac{2m}{n - m(n-2)}, \tag{4.11}$$

e $m = 0.15$, é um parâmetro que foi obtido através de testes em [16].

4.4.1 Questões de Convergência do Modelo Distribuído

De forma similar ao que ocorre no *Power Method*, este método apresenta problemas de convergência, pois as matrizes A_1, A_2, A_3 e A_4 não são irredutíveis. Isso ocorre devido a inserção de 1's nas diagonais, o que na prática implicaria no nó 4 só ter saída e chegada nele mesmo. Dessa forma, usando-se da matriz A_1, por exemplo, um navegador aleatório que partisse de 4, nunca sairia de lá.

Para isso é usada uma média no tempo, em que a cada iteração é feita uma média de todos os estados anteriores. Assim $y(k)$ é a média do conjunto de amostras $x(0), ..., x(k)$ definida como (4.12),

$$y(k) = \frac{1}{k+1} \sum_{l=0}^{k} x(l). \tag{4.12}$$

De tal forma que o algoritmo converge no sentido da média quadrada, ou seja,

$$\lim_{k \to \infty} \mathbb{E}[\| y(k) - x^* \|^2] = 0. \tag{4.13}$$

Neste contexto, quanto maior for o número de iterações k, mais o valor de $y(k)$ se aproxima da distribuição limite, o que representa o valor de *PageRank*.

A fim de minimizar custos computacionais, um formato recursivo da média $y(k)$ foi usado nas simulações, que possui o seguinte formato:

$$y(k+1) = \frac{(k+1)}{(k+2)} y(k) + \frac{1}{(k+2)} x(k+1). \tag{4.14}$$

A média em seu formato recursivo (4.14) diminui o custo computacional pois a cada nova iteração do vetor $y(k)$ apenas leva-se em conta o seu estado anterior $y(k-1)$. A média em seu formato comum (4.12) é estimada a cada nova iteração a partir de todos os outros estados anteriores, o que eleva consideravelmente o custo computacional. A dedução do modelo recursivo da média é apresentado no Apêndice A.

5 *Simulações*

Neste capítulo são apresentadas as simulações feitas dos modelos anteriormente apresentados. Na realização das simulações optou-se pelo uso da linguagem MatLab, por ser simples e conveniente em um primeiro momento, em que não seria necessário um grande poder de processamento. O código em Matlab das simulações encontra-se no Apêndice B. O exemplo usado nas simulações é de um conjunto de quatro páginas *Web*, ou seja, um grafo de quatro nós, ou ainda, uma cadeia de Markov com quatro estados. As simulações foram feitas a partir dos modelos descritos anteriormente e a partir do grafo da figura 5.1 com o objetivo de observar algumas considerações feitas na literatura.

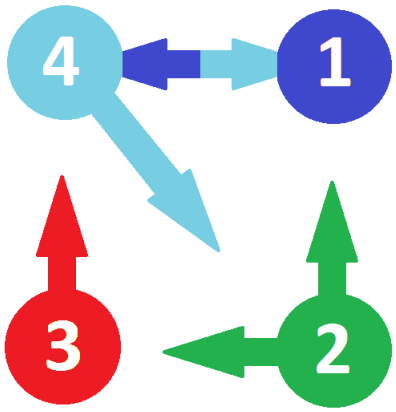

Figura 5.1: Grafo usado nas simulações.

Além disso, todas as simulações foram realizadas com base na seguinte condição inicial:

$$x_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \tag{5.1}$$

5.1 Simulação do *Power Method*

Na simulação do modelo mais simples do cálculo do *PageRank*, o *Power Method*, observa-se uma rápida convergência para cada um dos valores do vetor de estados. Isso ocorre devido a dimensão do problema usado na simulação. As cores das linhas dos gráficos, que representam os resultados das simulações, seguem o mesmo esquema das cores do grafo na figura 5.1. Desta forma, as linhas do gráfico estão associadas às páginas da seguinte forma:

- página 1: azul escuro,
- página 2: verde,
- página 3: vermelho,
- página 4: azul claro.

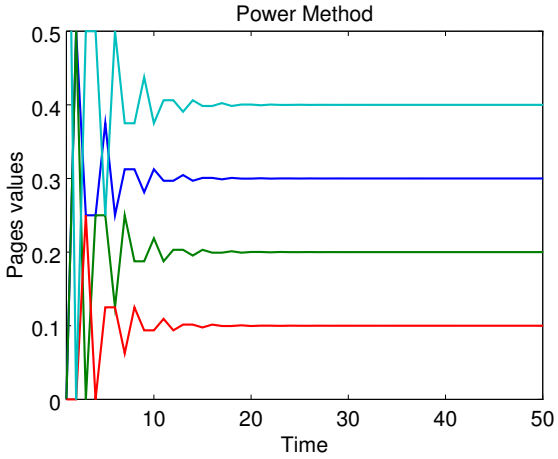

Figura 5.2: Resultado das simulações do *Power Method*.

Assim pode-se observar que a página de número 4 foi a que recebeu um maior *PageRank*, dentro deste conjunto de páginas. Ao mesmo que 4 é a página que está associada a um maior número de links, tanto de entrada quanto de saída, o que define seu alto valor de *PageRank*.

Entretanto, um resultado em primeira vista parece não ser conveniente ao grafo. Ao analisar-se o valor final de 1 observa-se que ele está na frente da página 2, e 2 possui mais *links* de saída que 1. Fazendo-se uma análise mais cuidadosa nota-se que 1 possui *links* de entrada e saída para 4, ou seja por estar ligada a 4 e por 4 possuir alto *PageRank*, o número de acessos a 1 foi maior por estar próximo a uma página supostamente importante.

5.2 Simulação do *Teleportation Model*

Na simulação do *Teleportation Model*, figura 5.3, também observa-se uma rápida convergência como nos resultados da simulação do *Power Method*.

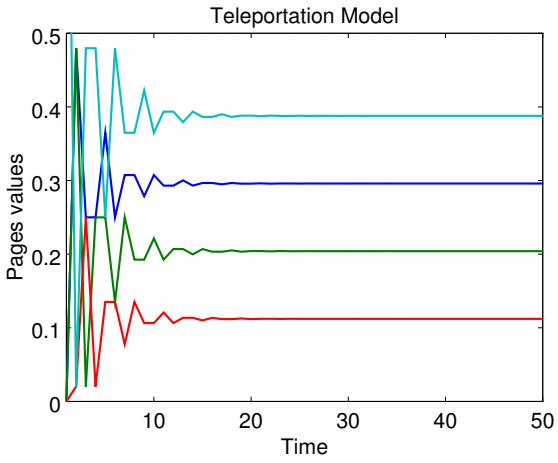

Figura 5.3: Resultado das simulações do *Teleportation Model*.

Além disso uma grande semelhança é observada entre as duas simulações, como pode-se constatar na figura 5.4. Tanto a semelhança quanto a rápida convergência se devem ao fato do exemplo simulado não possuir um enorme número de estados n e por ser bem comportado.

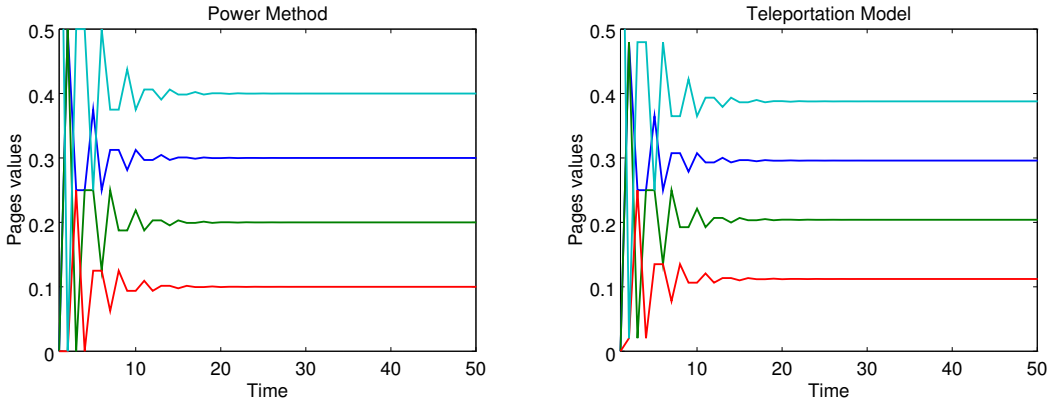

Figura 5.4: Comparação entre o *Power Method* e o *Teleportation Model*.

5.3 Simulações do Modelo Distribuído

As simulações do modelo distribuído foram divididas em duas subseções, uma com os resultados do modelo distribuído sem a média no tempo e outra com a média no tempo em sua forma recursiva. É possível observar que os resultados convergem com a média e não convergem sem ela.

5.3.1 Simulação do *Teleportation Model* Distribuído

No modelo distribuído sem a média no tempo observa-se uma oscilação dos estados sem que os valores das variáveis aleatórias atinjam a distribuição limite, ou seja, neste modelo sem a média no tempo não foi possível encontrar um valor de *PageRank* mesmo depois de 500 iterações. Os resultados dessa simulação estão apresentados na figura 5.5.

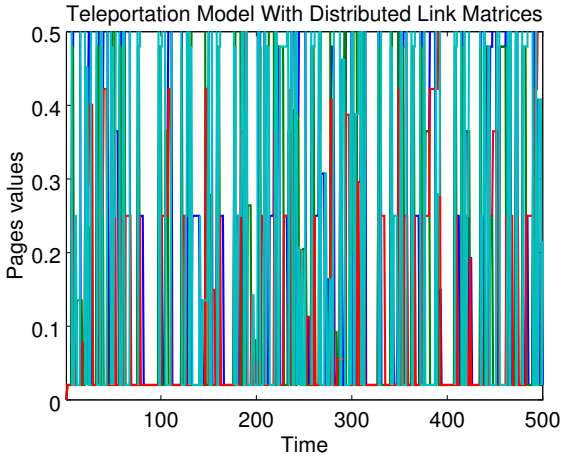

Figura 5.5: Resultado das simulações do modelo distribuído.

A não convergência ocorre devido as matrizes distribuídas possuirem 1's em suas diagonais, o que implicaria, por exemplo, usando-se a matriz A_1 como referência, nos nós 2, 3 e 4 só terem saídas para si mesmo, seguindo o esquema apresentado na figura 5.6.

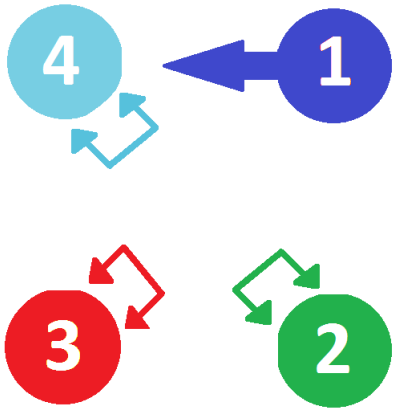

Figura 5.6: Abstração dos *links* das páginas 2, 3 e 4 a partir da matriz A_1.

5.3.2 O Modelo Recursivo da Média no Tempo Aplicada a Simulação do Modelo Distribuído

Entretanto, feita a média dos valores do modelo distribuído os resultados convergem, conforme é mostrado na figura 5.7. Mas ainda assim é possível observar algumas oscilações no gráfico, esta simulação também foi feita ao longo de 500 iterações.

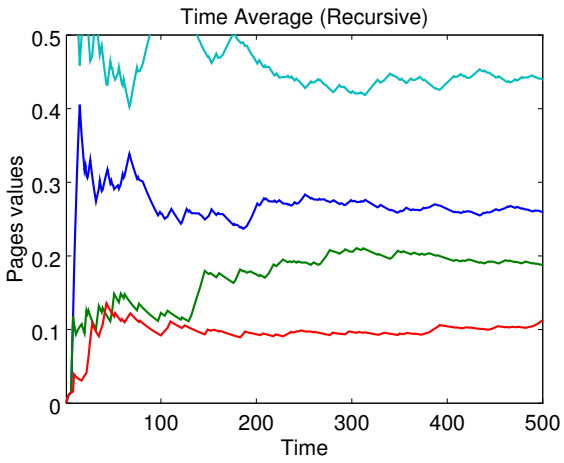

Figura 5.7: Resultado das simulações do modelo distribuído com a média no tempo.

Além disso, observa-se que na distribuição limite são atingidos valores bem próximos aos anteriormente encontrados para cada um dos estados no modelo do *Power Method*, o que pode ser constatado a partir da figura 5.8.

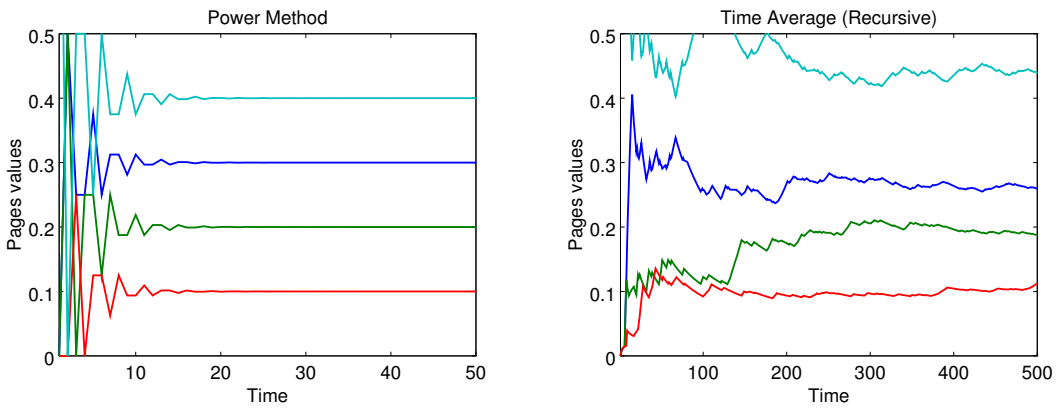

Figura 5.8: Comparação entre o *Power Method* e a média no tempo do modelo distribuído.

5.4 Método de *Monte Carlo* Aplicado após Modelo Recursivo da Média

Por fim, foi feita uma simulação usando método de Monte Carlo [1], no qual para cada uma das 500 iterações que podem ser vistas no gráfico foram feitas mais 500 iterações antes dos resultados serem plotados de forma a fazer uma média com os valores obtidos depois da média no tempo. Os resultados desta simulação encontram-se na figura 5.9.

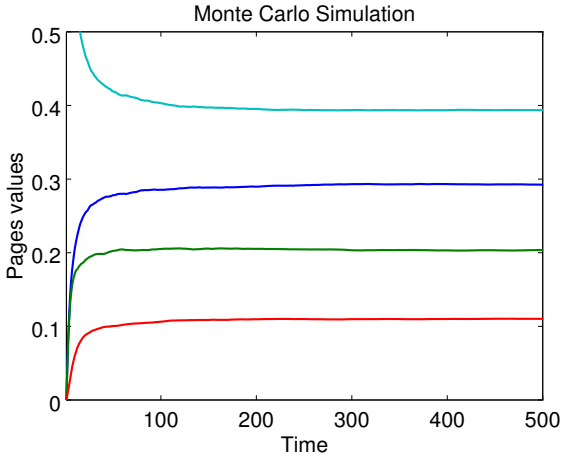

Figura 5.9: Resultado das simulações com o método de Monte Carlo.

Comparando a simulação em que usou-se do método de Monte Carlo com a simulação da média no tempo, observa-se que o gráfico do método de Monte Carlo possui uma curva mais suavizada, o que pode ser constatado nos resultados apresentados na figura 5.9. Além disso a evolução dos valores está mais bem definida, principalmente quando comparados ao valor final da distribuição limite.

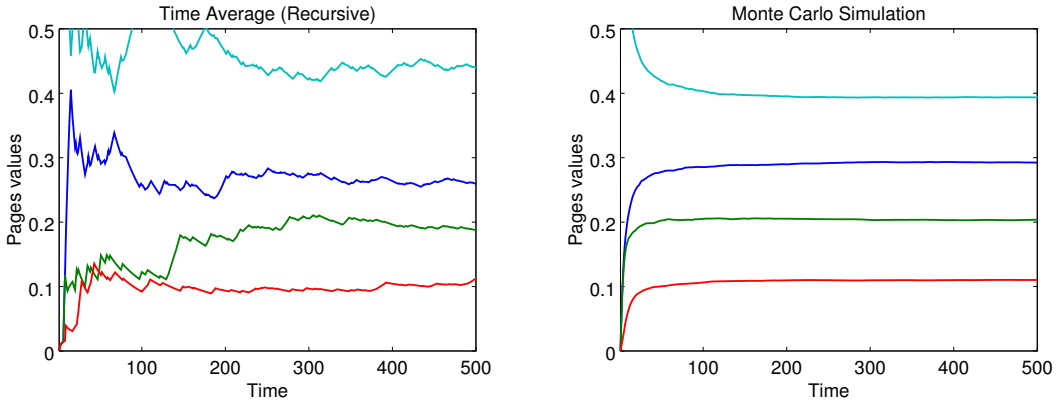

Figura 5.10: Comparação entre a média no tempo do modelo distribuído e o método de Monte Carlo.

E ainda comparando os resultados da simulação com Monte Carlo e da simulação do *Power Method* 5.11, nota-se que os valores finais estão de acordo com os alcançados no modelo do *Power Method*, o que sugere que o modelo é válido.

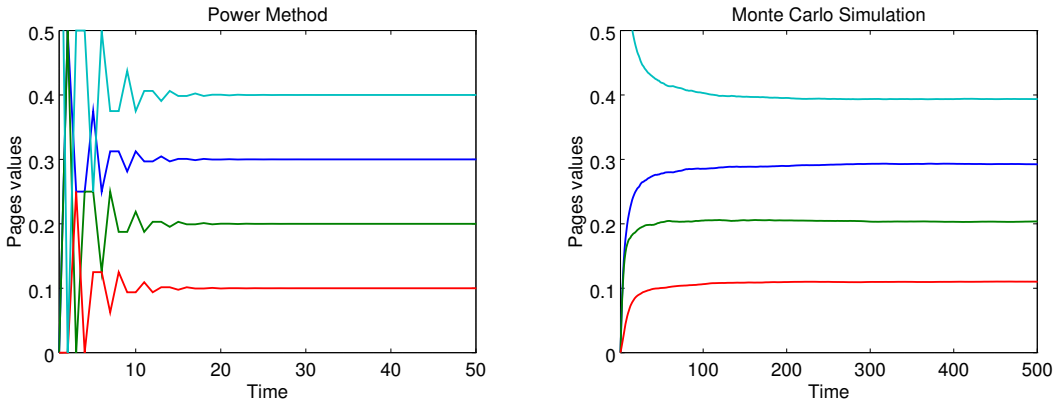

Figura 5.11: Comparação entre os resultados do *Power Method* e os obtidos com o método de Monte Carlo.

6 *Considerações Finais*

Desde o início das pesquisas em desenvolvimento de ferramentas de busca em 1994, conseguir fazer um ranqueamento de páginas *Web* de acordo com os interesses do usuário tem sido a parte mais desafiadora no desenvolvimento desses sistemas. Foi talvez por partir desse problema, e com uma boa solução, que o buscador Google obteve tamanho sucesso desde sua criação ao final da década de 90. As técnicas e modelos por trás do ranqueamento desse conjunto de documentos foram publicadas e, desde então, a comunidade científica tem trabalhado na pesquisa de novos modelos para tornar o cálculo do *PageRank* mais eficiente.

O cálculo do *PageRank* pode ser representado por diversos modelos, mas de forma geral é apresentado como uma equação de diferença, com algumas modificações na matriz de transição, a fim de tratar possíveis problemas de convergência. Os modelos e as propostas mais recentes do algoritmo estão voltados para o problema da distribuição do cálculo do ranquemento, no intuito de torná-lo cada vez mais eficiente. Mas a atenção ao algoritmo não tem por fim somente o ranqueamento de páginas *Web*, a ideia do cálculo do *PageRank* pode ser aplicada a diversos outros problemas.

Durante o trabalho foi realizada uma revisão dos modelos propostos para o algoritmo do *PageRank* na literatura. De forma que, inicialmente fosse efetuado um estudo dos modelos matemáticos por trás do algoritmo e em seguida fossem realizadas as simulações. Também para que fossem compreendidos os modelos matemáticos por trás do algoritmo, foi realizado um estudo sobre questões relacionadas a sistemas dinâmicos, sistemas estocásticos e probabilidade.

Neste trabalho ainda pretende-se utilizar outros métodos válidos na simulação do *PageRank*. Como a cada ano novos artigos são publicados no tema, novos modelos e técnicas estão sempre sendo publicados pela comunidade científica. Porém uma atenção maior seria dada a modelos agregados de cadeias de Markov, e problemas de consenso. E ainda como uma continuação deste trabalho, poderia ser feita uma implementação com *links* já coletados por um *Web Crawling*. De forma a por em prática os modelos num ambiente real e usando-se de técnicas de computação distribuída, afim de otimizar o cálculo e por em prática os algoritmos distribuídos.

Além de usar outras linguagens de programação, como C e C++, por questões de desempenho e por serem mais adequadas a implementações em *clusters*.

Apêndices

Apêndice A - Dedução do Modelo Recursivo da Média no Tempo

Partindo-se da equação da média no tempo (4.12), e fazendo-se um avanço de uma iteração no vetor da média, tem-se:

$$y(k) = \frac{1}{k+1}\sum_{l=0}^{k} x(l)$$

$$y(k+1) = \frac{1}{k+2}\sum_{l=0}^{k+1} x(l)$$

$$y(k+1) = \frac{1}{k+2}[x(0)+x(1)+...+x(k)+x(k+1)]$$

$$y(k+1) = \frac{1}{k+2}\sum_{l=0}^{k} x(l) + \frac{1}{k+2}x(k+1)$$

$$y(k+1) = \frac{1}{k+2}(k+1)y(k) + \frac{1}{k+2}x(k+1)$$

Por fim, o método recursivo da média no tempo é dado pela seguinte equação:

$$y(k+1) = \frac{k+1}{k+2}y(k) + \frac{1}{k+2}x(k+1)$$

Esta equação segue o mesmo formato da apresentada em (4.14).

Apêndice B - Código em Linguagem MATLAB Usado nas Simulações

```
%Simulacao do PageRank
%Faculdade de Educacao Tecnologica do Estado do Rio de Janeiro - FAETERJ
%Laboratorio Nacional de Computacao Cientifica - LNCC
%Oscar Neiva Eulalio Neto
%Orientador: Eduardo Krempser da Silva
%Co-orientador: Marcos Garcia Todorov

clear
clc
close all

%Numero de estados
N = 4;

%Numero de matrizes distribuidas
NPn = N;

%Horizonte de tempo
T = 50;

%Horizonte de tempo do metodo de monte carlo
itmax = 50;

%Parametro m do modelo distribuido
m = 0.15;

%Vetor aleatorio
%x = rand(1,N);
%x = diag(x*ones(N,1))\x;

%Vetores estaticos
x = [0 0 0 1];

%Outros vetores
y = x;
xd = x;
yd = x;
z = x;
```

```
zr = x;
w = zeros(T+1,N);

%Matriz estocastica aleatoria (linha)
%P = rand(N);
%P = diag(P*ones(N,1))\P;

%Matriz estocastica estatica
P = [0 0 0 1; 0.5 0 0.5 0; 0 0 0 1; 0.5 0.5 0 0];

%Vetor de uns
v1 = ones(1,N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Equacao de M
M = 2*m/(N-m*(N-2));

%Matriz tridimensional de zeros
PP = zeros(size(P,1),size(P,2),N);

%Insere valores na matriz tridimensional
for i=1:N
    %Insere uns nas diagonais
    PP(i,i,:) = 1;

    %Copia linhas para as matrizes distribuidas
    PP(i,:,i) = P(i,:);
end

for it=1:itmax
    for k=1:T
        %Gera numero para representar a matriz distribuida
        Pn = ceil(rand*NPn);

        %Power Method
        x(k+1,:) = x(k,:)*P;

        %Teleportation model (nao distribuido)
        y(k+1,:) = (1-M)*(x(k,:)*P) + (M/N)*v1;

        %Matrizes links distribuidas
        xd(k+1,:) = xd(k,:) * PP(:,:,Pn);
```

```
            %Teleportation model (distribuido)
            yd(k+1,:) = (1-M)*(xd(k,:)*PP(:,:,Pn)) + (M/NPn)*v1;

            %Media no tempo (nao recursiva)
            z(k,:) = sum(yd)/(T+1);

            %Media no tempo (recursiva)
            zr(k+1,:) = (((k+1)/(k+2)) * zr(k,:)) + ((1/(k+2))*yd(k+1,:));
        end
        w = w + zr./itmax;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Imprime resultados
clf

figure(1);
%subplot(211);
plot(x,'Linewidth',2.0);
set(gca,'fontsize',20);
set(gca,'XLim',[1 T]);
set(gca,'YLim',[0 2/N]);
title('Power Method');
ylabel('Pages values');
xlabel('Time');
print('-depsc2','powermethod');

figure(2);
%subplot(212);
plot(y, 'Linewidth', 2.0);
set(gca,'fontsize',20);
set(gca,'XLim',[1 T]);
set(gca,'YLim',[0 2/N]);
title('Teleportation Model');
ylabel('Pages values');
xlabel('Time');
print('-depsc2','teleportation');

%figure(3);
%subplot(211);
%plot(xd, 'Linewidth', 1.0);
```

```
%set(gca,'fontsize',20);
%set(gca,'XLim',[1 T]);
%set(gca,'YLim',[0 2/N]);
%title('PageRank With Distributed Link Matrices');
%ylabel('Pages values');
%xlabel('Time');
%print('-depsc2','pagedistributed');

figure(4);
%subplot(212);
plot(yd, 'Linewidth', 2.0);
set(gca,'fontsize',20);
set(gca,'XLim',[1 T]);
set(gca,'YLim',[0 2/N]);
title('Teleportation Model With Distributed Link Matrices');
ylabel('Pages values');
xlabel('Time');
print('-depsc2','teledistributed');

%figure(5);
%subplot(7,2,9);
%plot(z);
%set(gca,'fontsize',20);
%set(gca,'XLim',[1 T]);
%set(gca,'YLim',[0 2/N]);
%title('Time Average (Not Recursive)');
%ylabel('Pages values');
%xlabel('Time');
%print('-depsc2','polyak','-F:30');

figure(6);
%subplot(211);
plot(zr, 'Linewidth', 2.0);
set(gca,'fontsize',20);
set(gca,'XLim',[1 T]);
set(gca,'YLim',[0 2/N]);
title('Time Average (Recursive)');
ylabel('Pages values');
xlabel('Time');
print('-depsc2','timerecursive');

figure(7);
%subplot(212);
```

```
plot(w, 'Linewidth', 2.0);
set(gca,'fontsize',20);
set(gca,'XLim',[1 itmax]);
set(gca,'YLim',[0 2/N]);
title('Monte Carlo Simulation');
ylabel('Pages values');
xlabel('Time');
print('-depsc2','montecarlo');
```

Referências

[1] Konstantin Avrachenkov, Nelly Litvak, Danil Nemirovsky, and Natalia Osipova. Monte carlo methods in pagerank computation: When one iteration is sufficient. *SIAM Journal on Numerical Analysis*, 45(2):890–904, 2007.

[2] P. Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues*, volume 31 of *Texts in Applied Mathematics*. Springer, New York, 1999.

[3] Sergey Brin and Lawrence Page. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks*, 56(18):3825–3833, 2012.

[4] K. Bryan and T. Leise. The $25,000,000,000 eigenvector: The linear algebra behind Google. *SIAM Rev.*, 48(3):569–581, 2006.

[5] O. L. V. Costa, M. D. Fragoso, and R. P. Marques. *Discrete-Time Markov Jump Linear Systems*. Probability and Its Applications. Springer-Verlag, New York, 2005.

[6] O. L. V. Costa, M. D. Fragoso, and M. G. Todorov. *Continuous-Time Markov Jump Linear Systems*. Probability and Its Applications. Springer-Verlag, Heidelberg, 2013.

[7] U. Doraszelski and K. L. Judd. Avoiding the curse of dimensionality in dynamic stochastic games. *Quantitative Economics*, 3(1):53–93, 2012.

[8] O. Häggström. *Finite Markov Chains and Algorithmic Applications*. Cambridge University Press, Cambridge, 2002.

[9] T. Hou, W. Zhang, and H. Ma. Finite horizon H_2/H_∞ control for discrete-time stochastic systems with Markovian jumps and multiplicative noise. *IEEE Trans. Automat. Control*, pages 1185–1191, 2010.

[10] H. Ishii, R. Tempo, , and E.-W. Bai. PageRank computation via a distributed randomized approach with lossy communication. *Systems Control Lett.*, 61:1221–1228, 2012.

[11] H. Ishii, R. Tempo, , and E.-W. Bai. A web aggregation approach for distributed randomized PageRank algorithms. *IEEE Trans. Automat. Control*, 57(11):2703–2717, 2012.

[12] H. Ishii and R. Tempo. Distributed randomized algorithms for the PageRank computation. *IEEE Trans. Automat. Control*, 55(9):1987–2002, 2010.

[13] Hideaki Ishii and Roberto Tempo. The pagerank problem, multiagent consensus, and web aggregation: A systems and control viewpoint. *Control Systems, IEEE*, 34(3):34–53, 2014.

[14] Jianjun Lei and Han-Fu Chen. Distributed randomized pagerank algorithm based on stochastic approximation. 2015.

[15] Oliver A McBryan. Genvl and wwww: Tools for taming the web. In *Proceedings of the first international world wide web conference*, volume 341. Geneva, 1994.

[16] Nazar Zaki, Jose Berengueres, and Dmitry Efimov. Detection of protein complexes using a protein ranking algorithm. *Proteins: Structure, Function, and Bioinformatics*, 80(10):2459–2468, 2012.