UNIVERSIDADE FEDERAL DO PARANÁ
SETOR DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

RORYCKLES MIRANDA MELO KINAPE

# ANÁLISE DE UM MODELO PARA GERAÇÃO DE CARGA DE SERVIDORES WEB UTILIZANDO CLASSIFICAÇÃO DE CONTEÚDO

CURITIBA

2013

RORYCKLES MIRANDA MELO KINAPE

# ANÁLISE DE UM MODELO PARA GERAÇÃO DE CARGA DE SERVIDORES WEB UTILIZANDO CLASSIFICAÇÃO DE CONTEÚDO

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia Elétrica, Departamento de Engenharia Elétrica, Setor de Tecnologia, Universidade Federal do Paraná, como parte das exigências para a obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Carlos Marcelo Pedroso

CURITIBA

2013

**TERMO DE APROVAÇÃO**

RORYCKLES MIRANDA MELO KINAPE

ANÁLISE DE UM MODELO PARA GERAÇÃO DE CARGA DE SERVIDORES WEB UTILIZANDO CLASSIFICAÇÃO DE CONTEÚDO

Monografia aprovada como requisito parcial para a obtenção do grau de Bacharel em Engenharia Elétrica, Setor de Tecnologia da Universidade Federal do Paraná, pela seguinte banca examinadora:

Orientador: _____
Prof. Dr. Carlos Marcelo Pedroso
Departamento de Engenharia Elétrica, UFPR

_____
Prof. Dr. Evelio Martín García Fernández
Departamento de Engenharia Elétrica, UFPR

_____
Prof. Dr. Luis Henrique Assumpção Lolis
Departamento de Engenharia Elétrica, UFPR

Curitiba, 9 de agosto de 2013.

## RESUMO

A modelagem do tráfego *Web* atual tem importante papel no dimensionamento de servidores e previsão de desempenho destes. Neste trabalho é feita uma análise de um novo modelo para geração de carga de servidores, que possui como diferencial a classificação semântica de conteúdo dos arquivos transferidos, em comparação a um dos modelos mais utilizados atualmente. Para tal, foi necessário desenvolver ferramentas para parametrizar e simular sessões de usuários utilizando ambos os modelos em teste. A parametrização dos modelos foi realizada com base em um *trace*, que é uma amostra de tráfego real. Os resultados das simulações de cada modelo foram comparados com este *trace* no que diz respeito aos níveis de autocorrelação do tráfego agregado resultante em diversas escalas de tempo, uma vez que essa característica é importante para avaliação de desempenho deste tipo de sistema. Neste aspecto, é realizada uma comparação entre a função de autocorrelação observada no tráfego real e a função de autocorrelação observada no tráfego gerado utilizando o modelo proposto, bem como com um dos modelos mais utilizados na literatura.



Palavras-chave: Modelo. Tráfego *Web*. Autocorrelação. Cauda pesada.

## ABSTRACT

Web traffic modeling has important role on server dimensioning and performance evaluation. This monograph presents an analysis of a new model for workload generation of Web servers, based on classification of server files in groups, compared to one of the currently most used models. For this, it's been necessary to develop tools to parameterize and simulate user session of both tested models. The models parameters were set based on a real traffic trace. The results of simulations for each model have been compared with that trace concerning the autocorrelation of the size of transmitted files, as this feature is important to performance evaluation of Web systems. The autocorrelation function observed in real traffic was compared to synthetic traces generated by proposed model.



Keywords: Workload model. Web traffic. Autocorrelation. Heavy tail.

## LISTA DE ILUSTRAÇÕES

## LISTA DE GRÁFICOS

## LISTA DE TABELAS

## LISTA DE SIGLAS

**ACF** – *Autocorrelation Function* – Função de autocorrelação

**CDF** – *Cumulative distribution function* – Função de Distribuição Acumulada

**FTP** – *File Transfer Protocol* – Protocolo de Transferência de Arquivos

**HTTP** – *Hypertext Transfer Protocol* – Protocolo de Transferência de Hipertexto

**HTTPS** – *Secure Hypertext Transfer Protocol* – Protocolo de Transferência Segura de Hipertexto

**IP** – *Internet Protocol*

**MB** – Mega *bytes* – Um milhão de *bytes*

**NS-2** – *Network Simulator* 2

**PUCPR** – Pontifícia Universidade Católica do Paraná

**URL** – *Uniform Resource Locator* – Localizador Padrão de Recursos

**UTC** – *Universal Time Coordinated* – Tempo Universal Coordenado

**SUMÁRIO**

# 1 Introdução

O grande sucesso da *Internet* é demonstrado pelo seu crescimento exponencial e seu papel fundamental nos meios sociais, econômicos, entre outros. A comunidade acadêmica, contudo, percebeu que a estrutura atual da rede mundial de computadores possui limitações que impedem a sua evolução e desenvolvimento das necessidades dos usuários [2].

O tráfego *Web* baseado no protocolo HTTP ainda é a aplicação dominante na *Internet*. E devido às redes sociais, a páginas com vídeo embutido como o YouTube, e a *sites* de hospedagem de arquivos, o percentual do tráfego *Web* é significativo [4], podendo chegar a até 39% do tráfego total da *Internet* [2]. Com isso, faz-se necessário que o projeto da estrutura de rede e o dimensionamento de servidores sejam realizados de modo a atender a atual e a futura demanda. “Bons modelos permitem realizar previsões precisas sobre métricas de desempenho e, a partir destas, por exemplo, o planejamento da capacidade do servidor” [1].

Estudos anteriores observaram que o tráfego *Web* agregado possui grandes níveis de autocorrelação [20, 21], tornando difícil realizar o dimensionamento de *buffers* e previsão de desempenho analiticamente. Em alguns casos, somente simulações numéricas podem resolver este problema totalmente, enfatizando a importância de se utilizar modelos precisos.

Este trabalho apresenta uma análise de comparação entre os níveis de autocorrelação encontrados nas simulações de dois modelos de geração de tráfego *Web*: o primeiro foi proposto por Pedroso e Fonseca [1] em 2006 – chamado neste estudo de modelo A – e tem como principal diferença de estudos anteriores [2, 3, 4, 7], a classificação do conteúdo dos arquivos a serem transmitidos. Isto é, leva-se em consideração o fato de que os arquivos transmitidos possuem diferentes características quanto ao seu tamanho, sendo necessária, portanto, uma parametrização diferenciada para cada classe de arquivo existente no servidor *Web*.

O segundo modelo que este trabalho aborda é de autoria de Choi e Limb [7] e é um dos mais utilizados na atualidade, sendo inclusive usado para padronizações [4] – ao longo deste trabalho será referenciado como modelo B.

## 1.1 Objetivos

### 1.1.1 Objetivo Geral

Comparar a característica de autocorrelação dos dois modelos de representação de tráfego *Web*, propostos por Pedroso e Fonseca [1] e por Choi e Limb [7], é o objetivo geral deste trabalho. Ambos os modelos devem ser comparados com uma amostra de tráfego *Web* real.

### 1.1.2 Objetivos Específicos

Os principais objetivos específicos são listados abaixo:

- Estudar o fundamento dos principais modelos de representação de tráfego *Web* atualmente, em especial o modelo de Choi e Limb [7].
- Analisar e processar o *log* usado como amostra de tráfego real.
- Parametrizar empiricamente os modelos em prova com base no *log*.
- Criar *scripts* para realizar as simulações.
- Analisar os níveis de autocorrelação provenientes das simulações e do tráfego real.

## 1.2 Justificativa

Este trabalho tem grande importância para o autor, pois além de ser um desafio a ser concluído, seu tema é de interesse e relevância para a formação acadêmica e profissional. Além do conhecimento teórico adquirido, soma-se a experiência prática que o trabalho proporciona, visto que para o autor realizar a análise dos resultados provenientes de cada modelo, é necessário antes parametrizar tais modelos e criar *scripts* para gerar as simulações.

Ao se tratar de planejamento de capacidade de servidores, é importante que os modelos de desempenho usados sejam precisos o suficiente para que não haja

sub ou superdimensionamento do sistema [1]. Analisando o novo modelo, proposto por Pedroso e Fonseca, será possível verificar se com ele se consegue melhores resultados em representação do tráfego *Web*.

Um novo e mais eficiente modelo de geração de tráfego HTTP beneficiará primeiramente a comunidade acadêmica e, num segundo momento, a indústria e empresas do setor de telecomunicações.

## 1.3 Metodologia

Para o desenvolvimento deste trabalho, o autor cumpriu as seguintes etapas:

- Estudo sobre os modelos de geração de tráfego HTTP atuais
- Estudo sobre o *software* estatístico R
- Processamento do *trace* utilizado como referência ao tráfego real
- Desenvolvimento de *scripts* em linguagem PERL
- Parametrização dos modelos em teste
- Simulações
- Análise dos resultados

## 1.4 Estrutura da Dissertação

No Capítulo 2, a seguir, conceitos abordados neste trabalho são descritos, uma vez que a compreensão destes é de extrema importância para o total entendimento das etapas seguintes. Também são apresentados os principais modelos de representação de tráfego *Web* existentes na atualidade.

O Capítulo 3 relata a descrição do *trace* que foi utilizado e os métodos que autor utilizou para processamento dos dados, parametrização dos modelos e simulações.

A análise dos resultados é feita no Capítulo 4. Nesta seção o autor apresenta os dois níveis de comparação entre simulações provenientes dos modelos e o tráfego real.

O Capítulo 5 resume o trabalho com a conclusão e uma breve discussão sobre trabalhos futuros.

# 2 Conceitos Fundamentais

## 2.1 Definição de Modelo

Um modelo é uma representação de um sistema que se deseja adquirir um maior entendimento. Supõe-se que o funcionamento de modelos deve ser similar à entidade que é alvo de estudo, de modo a demonstrar a consistência de teorias científicas [15]. Um bom modelo possui como importante característica a simplicidade, obtida ao se restringir somente aos aspectos que influenciem significativamente no comportamento que se deseja analisar [1].

Modelos para representação de tráfego HTTP simulam as transferências de arquivos de um servidor a diversos clientes, de modo que são utilizados em ferramentas para realizar análise de desempenho de servidores *Web*.

Ao decorrer deste trabalho serão feitas menções a determinadas características de modelos de geração de tráfego *Web* e que devem ser compreendidas pelo leitor. São elas:

- Objeto principal: arquivo requisitado pelo usuário, que pode ou não conter referências a outros arquivos (objetos embutidos). Ao receber o objeto principal, o navegador *Web* utilizado pelo usuário iniciará automaticamente a transferência dos objetos embutidos, se existirem.
- Objeto embutido: arquivo referenciado em um objeto principal.
- Requisição: conjunto composto por objeto principal e todos seus objetos embutidos.
- Sessão: conjunto de requisições feitas por um único usuário. Em geral, entre as requisições há o intervalo em que o usuário está lendo o conteúdo da página.

## 2.2 Principais Modelos para Geração de Tráfego HTTP

Atualmente os modelos abaixo citados são considerados o estado da arte em geração de tráfego *Web* (ou HTTP), alguns sendo inclusive utilizados em padronizações e também referenciados em artigos correlatos.

### 2.2.1 O Modelo Surge

O modelo SURGE (*Scalable URL Reference Generator*), proposto por Barford e Crovella em 1998, é divido em duas categorias: um “equivalente de usuário” e distribuições de probabilidade.

O “equivalente de usuário” é a forma como o modelo representa o comportamento do usuário ao classificar o tráfego em dois estados: ON (ligado) e OFF (desligado) [3]. O estado ON representa o período em que arquivos de uma mesma requisição estão sendo transmitidos e o estado OFF é o período ocioso entre duas requisições, que indica o momento em que o usuário está visualizando o conteúdo da página atual e está pensando na sua próxima ação, seja ela carregar uma nova página ou arquivo, ou simplesmente entrar em estado de inatividade.

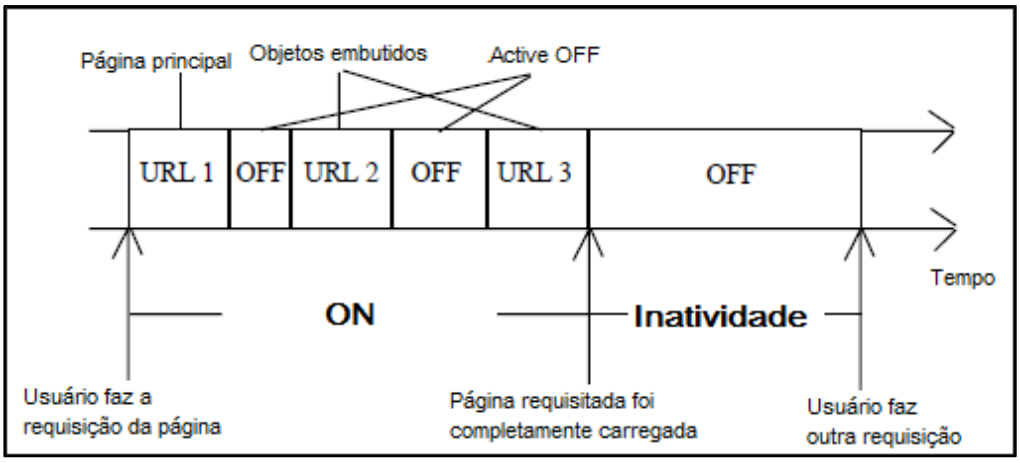



FIGURA 1 – MODELO SURGE: CARACTERIZAÇÃO DO COMPORTAMENTO DO USUÁRIO ADAPTADO DE: BARFORD E CROVELLA (1998)

Barford e Crovella levam em consideração em seu modelo o fato de que ao se carregar uma página, outros objetos serão consequentemente carregados. Por

exemplo, uma página HTML além de conter texto e *hyperlinks* – ligação entre páginas – pode conter imagens, vídeos e outros objetos embutidos. Portanto, ao se carregar a página HTML inicial, o navegador *Web* utilizado pelo usuário interpreta a existência desses objetos e em sequência transfere-os.

Contudo, há pequenos intervalos entre essas transferências de arquivos provenientes da requisição de uma página, chamados de *active-off*. Estes intervalos são causados pelo tempo em que o navegador leva para processar a página principal, pelas características da rede local e infraestrutura e funcionamento da *Internet*, entre outros [3]. Em geral, os intervalos de *active-off* são significativamente mais curtos que os intervalos de OFF (usuário inativo ou lendo o conteúdo da página).

Para caracterizar o tamanho dos arquivos contidos no servidor, Barford e Crovella fazem uso de distribuições de probabilidade, algumas das quais são caracterizadas como distribuições de cauda pesada, também conhecidas como distribuições de cauda longa, e que tem como característica a não convergência do desvio padrão.

O modelo de Barford e Crovella utiliza a mesma distribuição de probabilidade para todos os arquivos transmitidos.

### 2.2.2 O Modelo de Choi eLimb

Publicado em 1999, o artigo intitulado “*A Behavioral Model of Web Traffic*”, de Hyoung-Kee Choi e John Limb apresenta um modelo, referenciado ao longo deste trabalho como modelo B, que representa o comportamento do usuário e, em vez de tratar de páginas individuais, aborda requisições – que inclui uma página e seus objetos embutidos.

O estudo realizado por Choi e Limb foi feito com base em um *trace* coletado na rede *backbone* do campus do Instituto de Tecnologia da Geórgia (Georgia Institute of Technology), em Atlanta.

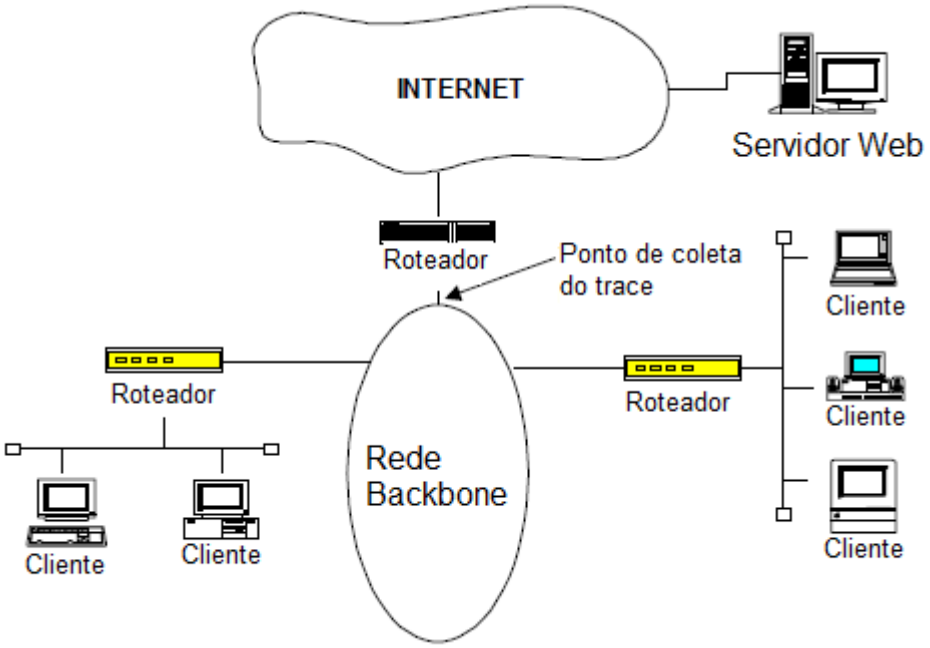



FIGURA 2 – TOPOLOGIA DA REDE DO INSTITUTO DE TECNOLOGIA DE GEÓRGIA
ADAPTADO DE: CHOI E LIMB (1999)

Como a rede da Universidade contém servidores *Web* que recebe acessos de computadores de fora da rede local, Choi e Limb tiveram que filtrar o trace para que fossem capturados somente os acessos da rede local com destino à *Internet* e não o contrário.

A partir deste *trace*, Choi e Limb então modelaram empiricamente distribuições estatísticas que representam as características do comportamento do usuário – tempo de OFF (usuário lendo o conteúdo da página) e número de requisições feitas pelo usuário durante uma sessão – e as características de requisição: tamanhos dos objetos principal e embutidos e tempo de *active-off*.

Para realizar as simulações, utilizou-se o fluxograma exibido na Figura 3. Ele representa a geração de cada requisição e pode ser interpretado da seguinte maneira: no início um objeto principal é transferido; em seguida a quantidade de objetos embutidos ao objeto principal é definida; um tempo de *active*-off é determinado para cada objeto embutido transferido; após todos os objetos embutidos serem transferidos, um tempo de OFF é definido e a requisição encerra; inicia-se uma nova requisição e o ciclo se repete.

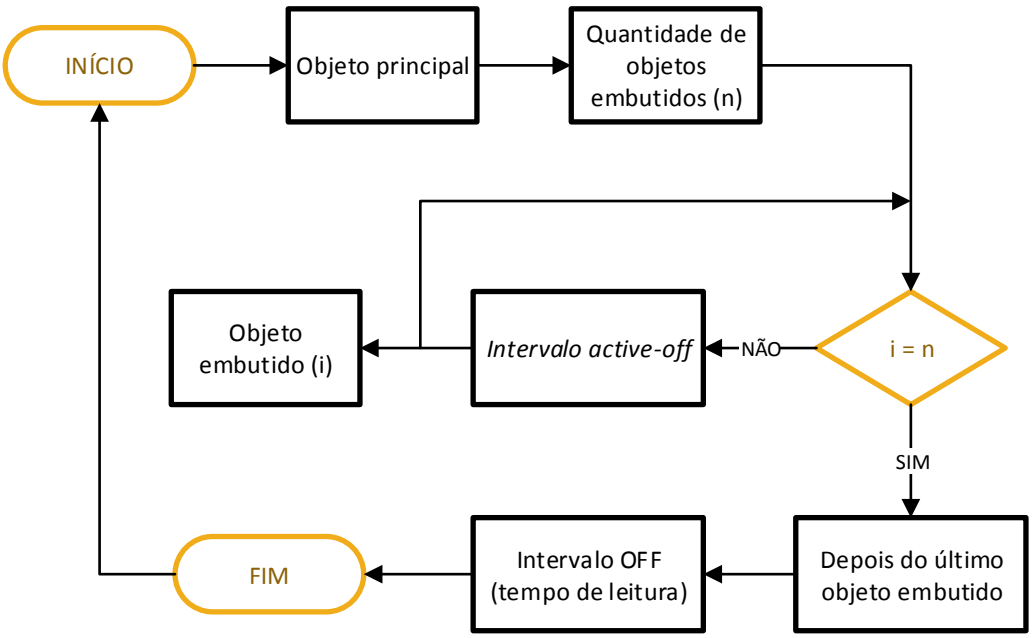



FIGURA 3 – FLUXOGRAMA DE TRANSIÇÃO DE ESTADOS DO MODELO B
ADAPTADO DE: CHOI E LIMB (1999)

Como resultado do estudo, Choi e Limb verificaram que existem níveis consideráveis de autocorrelação no tamanho dos objetos. A autocorrelação é uma medida que informa o quanto o valor de uma realização de uma variável aleatória é capaz de influenciar seus vizinhos. No caso do tamanho dos objetos, isto diz o quanto um objeto com tamanho grande condiciona os objetos em sequência a possuírem um tamanho também grande e vice-versa. O valor de autocorrelação pode variar de 1 (correlação perfeita) a -1 (anticorrelação), onde 0 indica a não existência de correlação.

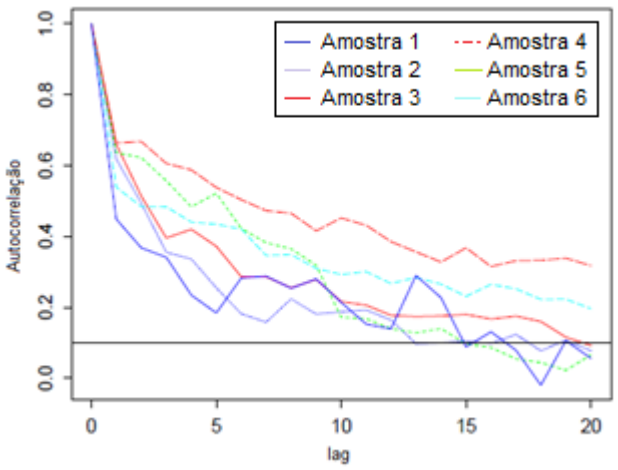



FIGURA 4 – AUTOCORRELAÇÃO ENCONTRADA NO TAMANHO DOS OBJETOS.
SEIS AMOSTRAS DE SIMULAÇÃO ESTÃO REPRESENTADAS.
ADAPTADO DE: CHOI E LIMB (1999)

## 2.3 O Modelo em Teste

O principal alvo de estudos deste trabalho é o modelo proposto por Carlos Marcelo Pedroso e Keiko Fonseca em 2006. O modelo A, como é chamado neste estudo, é baseado no modelo SURGE (abordado na seção 2.2.1) no que diz respeito ao comportamento do usuário (sistema ON-OFF), mantendo as mesmas características como intervalo de *active-off* e número de requisições em cada sessão.

O grande diferencial do modelo A, contudo, é a classificação semântica do conteúdo dos objetos. Pedroso e Fonseca observaram que atualmente os objetos transferidos possuem diferença considerável quanto ao seu tamanho dependendo da sua classe. Este modelo, portanto, difere das abordagens existentes, propondo o agrupamento dos objetos transmitidos pelo servidor Web em classes. Isto resulta em uma modelagem precisa do comportamento do sistema que permite a geração de tráfego sintético para simulações [1].

O modelo de Pedroso e Fonseca é constituído por sessões de vários usuários, cada qual contendo diversas requisições. Cada requisição é composta por um arquivo inicial podendo ser seguido de vários arquivos embutidos. Contudo, não é feita distinção entre arquivo inicial e arquivos embutidos, da mesma forma que não se usa uma distribuição estatística referente a um ou outro. Ao invés disso, o modelo A determina que no início de uma requisição existe uma probabilidade do arquivo inicial ser de determinada classe, e que existe outra probabilidade do arquivo seguinte ser da mesma classe ou de ainda outra.

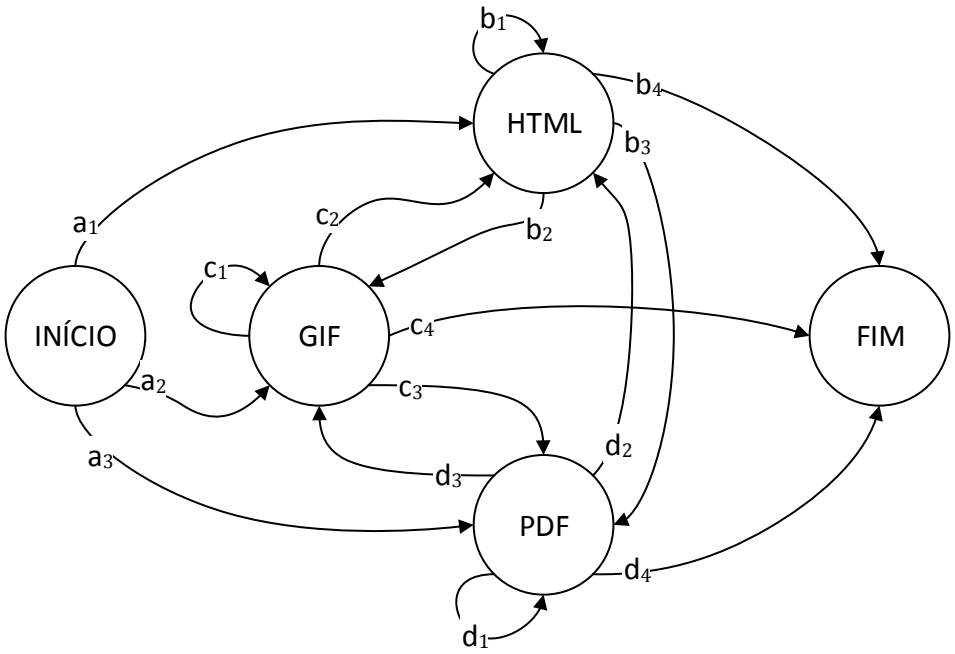



FIGURA 5 – DIAGRAMA HIPOTÉTICO DE CLASSES SEMÂNTICAS PARA TRANSMISSÃO DE OBJETOS DURANTE ESTADO ON
ADAPTADO DE: PEDROSO E FONSECA (2006)

A Figura 5 exemplifica como a classificação do conteúdo é abordada pelo modelo A. Ela pode ser interpretada da seguinte forma: uma requisição tem probabilidade $a_1$ de ser iniciada com uma página pertencente à classe HTML, que por sua vez será seguido por uma transmissão de outro arquivo. Neste caso $b_1$, $b_2$, e $b_3$ representam as probabilidades do próximo arquivo pertencer às classes HTML, GIF e PDF, e $b_4$ é a probabilidade de não haver transmissão de um novo arquivo, encerrando a requisição. A relação $\sum_{i=1}^{3} a_i = 1$ deve ser respeitada, assim como $\sum_{i=1}^{4} b_i = 1$, $\sum_{i=1}^{4} c_i = 1$ e $\sum_{i=1}^{4} d_i = 1$.

Esse diagrama de transição entre classes pode ser representando por uma matriz quadrada, exibida na Tabela 1.

TABELA 1 – MATRIZ DE PROBABILIDADE DE TRANSIÇÃO DE CLASSES

| Classe | HTML | GIF | PDF | Fim |
|---|---|---|---|---|
| Início | $a_1$ | $a_2$ | $a_3$ | 0 |
| HTML | $b_1$ | $b_2$ | $b_3$ | $b_4$ |
| GIF | $c_1$ | $c_2$ | $c_3$ | $c_4$ |
| PDF | $d_1$ | $d_2$ | $d_3$ | $d_4$ |

ADAPTADO DE: PEDROSO E FONSECA (2006)

O tempo de permanência em cada classe é o tempo de *active-off*, já mencionado em estudos anteriores. Pedroso e Fonseca identificaram que esse intervalo pode ser modelo por uma distribuição de Weibull.

A parametrização dos atributos do modelo A realizadas no presente trabalho é descrita na seção 3.3.1.2.

# 3 Material e Métodos

A seguir serão apresentadas descrições do *trace* utilizado como amostra de tráfego real, da parametrização dos modelos em teste e método utilizado para realização das simulações.

## 3.1 Descrição do Trace

O *log* utilizado como referência de tráfego real da *Internet* foi coletado em um servidor *Proxy* da Pontifícia Universidade Católica do Paraná (PUCPR) em 31 de Agosto de 2009 e contém o histórico de transferências de diversos arquivos feitas pelos alunos, professores e demais funcionários que utilizaram a rede de computadores da universidade. Servidores *Proxy* têm como finalidade intermediar os acessos de uma rede local e pequena a uma rede maior, como a *Internet* [6]. No caso da PUCPR, este intermédio do *Proxy* é utilizado para controlar e filtrar acesso dos usuários a conteúdos indesejados. Como o tráfego da *Internet* proveniente da rede local da universidade é centralizado neste servidor, foi possível armazenar o histórico de transferências, que é utilizado neste trabalho.

O *Proxy* originador do *log* utilizado neste trabalho é o Squid, uma plataforma amplamente utilizada que suporta HTTP, HTTPS, FTP e outros protocolos [12]. A função de *cache* do Squid não foi utilizada, isto é, o conteúdo das páginas e arquivos mais acessados não foi armazenado no servidor e todas as requisições foram encaminhadas para a *Web*.

O *trace* utilizado está contido em um único arquivo de 571 MB, que capturou requisições feitas a 36.959 *sites* diferentes durante um período de aproximadamente

9 horas e 38 minutos, totalizando 3.226.336 transferências. Para preservar a identidade dos usuários, o nome de *login* referente a cada transferência de arquivo no *log* foi substituído pela palavra “usuário”.

O *log* é formado por 3.226.336 linhas – uma transferência de arquivo por linha – e nove colunas, sendo elas:

1) Data: indicada a data em que a transferência foi feita; é representada no formato *Unix epoch*, que indica a quantidade de segundos que se passou desde a data de 1º de Janeiro de 1970 às 00:00:00 UTC [8].
2) Tempo, em milissegundos, de duração da transferência.
3) Endereço IP proveniente do computador originador.
4) Bloqueio do *Proxy*: indica se a transferência do arquivo foi bloqueada pelo servidor *Proxy* – indicado por “TCP_DENY” – ou se foi autorizada – indicado por “TCP_MISS”.
5) Tamanho do arquivo, em *Bytes*.
6) Método da requisição
7) Endereço URL do arquivo
8) Usuário: este campo originalmente continha o nome de *login* do usuário que requisitou a transferência, sendo substituído pela palavra “usuário” antes do *trace* ser disponibilizado.
9) Tipo de conteúdo

Um fragmento do *log* ilustrando o formato descrito acima é exibido na Figura 6.

```
1251695137.528    175 10.132.1.34 TCP_MISS/200 7521 GET http://www.google.com.br/ usuario text/html
1251695232.911      1 10.130.32.25 TCP_DENIED/407 2468 GET http://www.google.com.br/ usuario NONE/- text/html
1251695232.955      2 10.130.32.25 TCP_DENIED/407 2634 GET http://www.google.com.br/ usuario NONE/- text/html
1251695233.135    179 10.130.32.25 TCP_MISS/200 7521 GET http://www.google.com.br/ usuario text/html
```

FIGURA 6 – FRAGMENTO DO *LOG*
FONTE: O AUTOR (2013)

## 3.2 Processamento do Trace

O *log* acima descrito precisou ser processado em diversos momentos para que o seu uso fosse possível. Para tal, foram desenvolvidos diversos *scripts* em

linguagem Perl, uma linguagem de programação interpretada – ao invés do código ser compilado, ele é executado diretamente por um interpretador – amplamente utilizada para processamento de texto, entre outras finalidades [17].

Ambos os modelos de geração de tráfego *Web* estudados neste trabalho tem como objetivo possibilitar a análise de desempenho de servidores, de modo que o tráfego sintetizado pelo modelo represente as transferências realizadas entre vários usuários e um mesmo servidor.

Contudo, o *trace* coletado no *proxy* da PUCPR contém requisições feitas por diversos usuários a diversos servidores (*sites*), fato esse que é explicado pelo local onde o *trace* foi coletado. O *proxy* em questão é intermédio entre a rede local da universidade e a *Internet*. Soma-se a isso o comportamento dos usuários, onde há diferença entre as páginas acessadas entre eles, sendo que um único usuário pode ainda realizar acessos a vários *sites* diferentes.

Para lidar com essa característica do *log*, é necessário dividir o arquivo original em arquivos menores, cada qual contendo as requisições feitas a um único servidor. Inicialmente o autor verificou quais os servidores que possuem maior número de transferências, para dentre destes escolher quais serão utilizados no trabalho. O *script* desenvolvido e utilizado nesta etapa encontra-se no APÊNDICE A. Este *script* verifica cada linha do arquivo original, que corresponde a uma transferência, identifica o endereço (*host*) do servidor e armazena-o em uma tabela de um banco de dados MySQL. Ao término, fez-se a classificação da tabela SQL pela ordem decrescente da quantidade de aparições de cada servidor.

Na Tabela 2 encontra-se o resultado dessa análise. Devido à grande quantidade de servidores acessados (36.959), somente os vinte servidores mais acessados são exibidos.

TABELA 2 – SERVIDORES COM MAIOR NÚMERO DE TRANSFERÊNCIAS

| Servidor | Número de transferências |
|---|---|
| swupmf.adobe.com | 72296 |
| mail.google.com | 53821 |
| google.com.br | 53367 |
| google-analytics.com | 52676 |
| globo.com | 47047 |
| icisaude.org.br | 39739 |
| clients1.google.com.br | 38405 |
| stf.terra.com.br | 32835 |

continua

TABELA 2 – SERVIDORES COM MAIOR NÚMERO DE TRANSFERÊNCIAS

continuação

| Servidor | Número de transferências |
|---|---|
| l.yimg.com | 31153 |
| row.bc.yahoo.com | 30025 |
| sdp.terra.com.br | 28422 |
| br.adserver.yahoo.com | 25667 |
| googleads.g.doubleclick.net | 24194 |
| col.stb.s-msn.com | 23447 |
| gfx1.hotmail.com | 20781 |
| ad.yieldmanager.com | 20217 |
| rad.msn.com | 20118 |
| google.com | 19067 |
| globoesporte.globo.com | 18652 |
| h.msn.com | 18409 |

FONTE: O AUTOR (2013)

Dentre os servidores mais acessados, há alguns que não foram utilizados neste trabalho, como por exemplo, os servidores “swupmf.adobe.com” e “google-analytics.com”, pois não representam acessos provenientes de usuários, mas sim requisições feitas automaticamente por determinados *softwares* para verificar se há atualização de versão disponível, enviar estatísticas, entre outros.

Assim, o autor selecionou dentre a lista nove servidores que figuram como os mais frequentemente acessados pelos usuários desta rede, sendo assim compatíveis para serem usados neste estudo. São eles:

- globoesporte.globo.com
- l.yimg.com
- mail.google.com
- sdp.terra.com.br
- stf.terra.com.br
- www.globo.com
- www.google.com
- www.google.com.br
- www.icisaude.org.br

Com os servidores selecionados, a próxima etapa foi separar o arquivo de *log* original em arquivos contendo somente as requisições a estes *sites*. Para tal, o autor utilizou-se do Grep, uma ferramenta de linha de comando de sistemas Unix/Linux

que filtra um arquivo contendo texto e retorna somente as linhas que respeitem o termo desejado [18]. A sintaxe para execução dessa ferramenta é:

*grep <termo desejado> <arquivo>*

Juntamente com a ferramenta *grep*, foi utilizado o operador “>”, disponível em sistemas Unix/Linux, que faz com que o resultado seja gravado em um novo arquivo. A linha de comando utilizada para servidor é a seguinte:

*grep servidor logsProxy > logs-servidor*

Onde *servidor* foi substituído por cada um dos nove servidores, *logsProxy* é o arquivo contendo o *trace* completo (original) e *logs-<servidor>* é o arquivo resultante contendo somente as transferências para o servidor em questão.

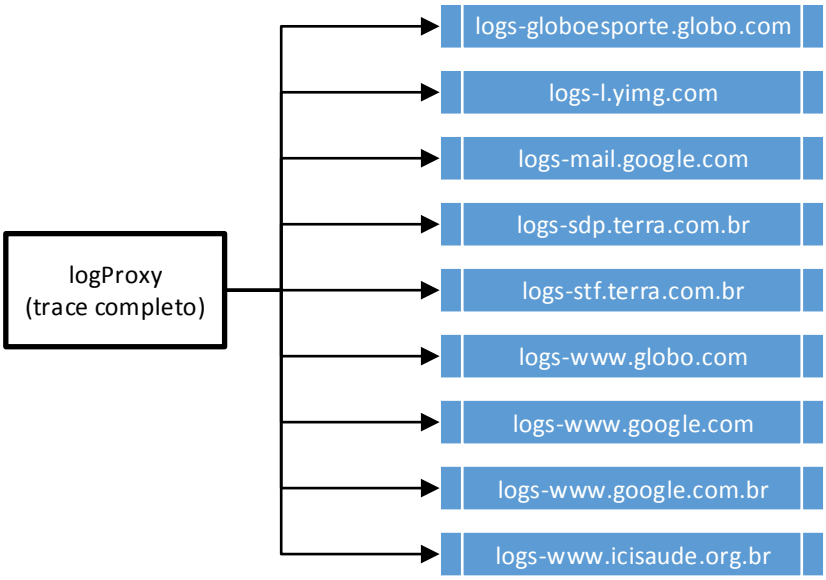



FIGURA 7 – FLUXOGRAMA DA SEPARAÇÃO DO *LOG* COMPLETO EM *LOGS* DE CADA SERVIDOR
FONTE: O AUTOR (2013)

A próxima etapa foi separar cada *log* de servidor em sessões, para que tenhamos em cada novo arquivo gerado as requisições feitas por cada usuário. Isto se fez necessário para parametrizar o comportamento do usuário para ambos os modelos em teste. O modelo A tem como mecanismo diferenciador a matriz de transição, que representa a probabilidade de determinada classe de arquivo ser transferida dada a classe do arquivo transferido anteriormente, e para modelar esta matriz também é necessário fazer a análise das sessões.

Para criar a matriz de transição também é necessário agrupar os tipos de arquivos em classes. Para esta finalidade foi desenvolvido o *script* que é

apresentado no APÊNDICE C. Este *script* lê cada um dos nove arquivos de *trace* dos servidores escolhidos e armazena em uma tabela SQL a quantidade que cada tipo de arquivo foi transferido. Conforme descrito na seção 3.1, a coluna nove do *trace* possui esta informação. Todavia, há casos de transferência de arquivo onde essa informação não está presente e o valor desta coluna é apresentando com um traço, conforme exemplo exibido na Figura 8.


FIGURA 8 – TRANSFERÊNCIA DE ARQUIVO ONDE NÃO HÁ INFORMAÇÃO DIRETA SOBRE O CONTEÚDO
FONTE: O AUTOR (2013)

Para estes casos particulares, o *script* verifica qual a extensão do arquivo pelo endereço URL, na sétima coluna. No exemplo acima, a extensão presente na URL é “gif”. Se na URL não houver informações sobre a extensão do arquivo, casos onde geralmente o usuário está abrindo a página principal do *site* através de sua página *default* (padrão) por exemplo, http://www.globo.com, o *script* então considera que o arquivo a ser transmitido é da classe HTML.

Na Tabela 3 é exibido o resultado produzido pelo *script*, listando o número de transferências que cada classe recebeu em cada servidor.

TABELA 3 – QUANTIDADE DE TRANSFERÊNCIAS POR CLASSE REALIZADAS NOS NOVE SERVIDORES SELECIONADOS

| CONTEÚDO | SITES | | | | | | | | | TRANSFERÊNCIAS |
|---|---|---|---|---|---|---|---|---|---|---|
| | globoesporte.globo.com | l.yimg.com | mail.google.com | sdp.terra.com.br | stf.terra.com.br | www.globo.com | www.google.com | www.google.com.br | www.icisaude.org.br | |
| html | html | html | html | html | html | html | html | html | html | 124837 |
| gif | gif | gif | gif | gif | gif | gif | gif | gif | gif | 57140 |
| js | js | js | js | - | js | js | js | js | js | 23632 |
| jpeg | jpeg | jpeg | jpeg | jpeg | jpeg | jpeg | jpeg | jpeg | - | 23380 |
| jpg | jpg | jpg | jpg | jpg | jpg | jpg | jpg | jpg | jpg | 21953 |
| css | css | css | css | css | css | css | css | css | css | 19530 |
| png | - | png | png | png | png | png | png | png | png | 16617 |
| javascript | - | - | javascript | - | - | - | javascript | javascript | javascript | 9727 |
| jsp | - | - | - | - | - | - | - | - | jsp | 7445 |
| x-javascript | x-javascript | x-javascript | x-javascript | - | x-javascript | x-javascript | x-javascript | - | - | 6243 |
| swf | swf | swf | swf | - | - | - | - | - | - | 2642 |
| plain | - | - | plain | - | - | - | plain | plain | - | 2603 |
| cur | - | - | cur | - | - | - | - | - | - | 1441 |
| x-shockwave-flash | x-shockwave-flash | x-shockwave-flash | x-shockwave-flash | - | x-shockwave-flash | - | - | - | - | 697 |
| json | - | - | - | - | - | - | json | json | - | 195 |
| htm | - | - | - | - | - | htm | - | htm | htm | 168 |
| jar | - | - | - | - | - | - | - | - | jar | 105 |
| octet-stream | - | - | octet-stream | - | - | - | - | - | - | 60 |
| vnd.ms-powerpoint | - | - | vnd.ms-powerpoint | - | - | - | - | - | - | 47 |
| msword | - | - | msword | - | - | - | - | - | - | 47 |
| pdf | - | - | pdf | - | - | - | - | pdf | pdf | 40 |
| x-ms-wmv | - | - | x-ms-wmv | - | - | - | - | - | - | 38 |
| bmp | bmp | bmp | bmp | - | - | - | - | - | - | 33 |
| vnd.ms-excel | - | - | vnd.ms-excel | - | - | - | - | - | - | 13 |
| dct | - | - | - | - | - | - | - | dct | - | 11 |
| class | - | - | - | - | - | - | - | - | class | 9 |
| mpeg | - | - | mpeg | - | - | - | - | - | - | 8 |
| x-zip-compressed | - | - | x-zip-compressed | - | - | - | - | - | - | 5 |
| php | - | - | - | - | - | - | - | php | - | 4 |
| zip | - | - | zip | - | - | - | - | - | - | 3 |
| vnd.oasis.opendocument.text | - | - | vnd.oasis.opendocument.text | - | - | - | - | - | - | 2 |
| vnd.dwg | - | - | vnd.dwg | - | - | - | - | - | - | 2 |
| xhtml+xml | - | - | - | - | - | - | xhtml+xml | - | - | 2 |
| asp | - | - | - | - | - | - | - | asp | - | 2 |
| dll | - | - | - | - | - | - | - | - | dll | 2 |
| x-icon | - | - | - | - | - | - | - | x-icon | - | 1 |
| aspx | - | - | - | - | - | - | - | aspx | - | 1 |

FONTE: O AUTOR (2013)

Através desse resultado, é possível observar que alguns tipos de arquivos são comuns aos nove servidores, como por exemplo, HTML e GIF, enquanto que outros tipos de arquivo, como PDF, estão presentes em poucos servidores. Outro aspecto observado é que certos tipos de arquivos podem ser agrupados, pois apesar de possuírem extensão divergente (JS, JAVASCRIPT e X-JAVASCRIPT, por exemplo), pertencem à mesma classe.

Assim, as oito classes definidas pelo autor são:

- HTML: referente a arquivos do tipo HTML e HTM
- GIF
- JPG: referente a arquivos do tipo JPG e JPEG
- JAVASCRIPT: referente a arquivos do tipo JS, JAVASCRIPT e X-JAVASCRIPT
- CSS
- PNG
- SWF: referente a arquivos do tipo SWF e X-SHOCKWAVE-FLASH
- OUTROS: referente aos demais tipos de arquivos que não se encaixam nas classes acima

A fim de separar os *traces* de cada um dos nove servidores em arquivos contendo sessões individuais de cada usuário, e já classificando cada tipo de arquivo de acordo com a sua classe atribuída, foi desenvolvido o *script* apresentado no APÊNDICE D.

Uma vez que o nome de usuário (oitava coluna) foi oculto do trace para por questões de privacidade, este *script* identifica as sessões através do endereço IP do computador que fez a requisição. Após definir a qual sessão pertence a transferência analisada, o *script* verifica em qual das oito classes o tipo de arquivo se encaixa. A Figura 9 exibe o fragmento de um dos arquivos de sessão gerados. O fluxograma da Figura 10 ilustra o funcionamento deste *script*.

```
1251718125.849 2622 html
1251718125.857 2456 html
1251718128.307 2489 outros
1251718130.339 2763 javascript
1251718130.339 2784 css
```

FIGURA 9 – FRAGMENTO DE UM ARQUIVO DE SESSÃO GERADO PELO SCRIPT, COM AGRUPAMENTO DE CLASSES JÁ REALIZADO

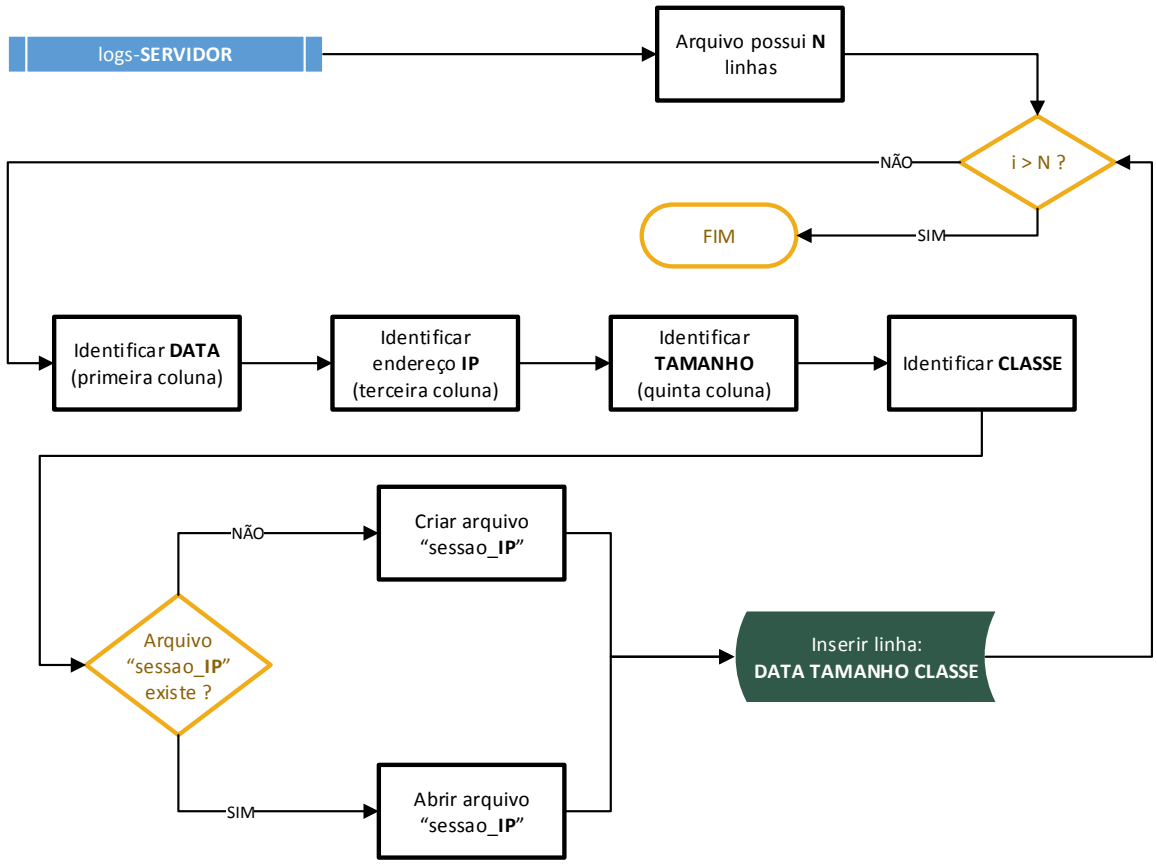



FIGURA 10 – FLUXOGRAMA DO SCRIPT PARA SEPARAR OS LOGS DE CADA SERVIDOR EM ARQUIVOS CONTENDO SESSÕES DE ÚNICO USUÁRIO, AGRUPANDO AS CLASSES
FONTE: O AUTOR (2013)

## 3.3 Parametrização

Os modelos A e B foram parametrizados a partir do *trace* utilizado neste trabalho como referência a tráfego real. Nesta seção serão descritos os métodos utilizados e parâmetros alcançados.

### 3.3.1 Parâmetros do Modelo A

#### 3.3.1.1. Matriz de Probabilidade de Transição

A primeira característica que deve ser parametrizada do modelo A é a matriz de probabilidade de transição. Para isso, deve-se fazer uma análise estatística em

cada sessão de cada servidor para verificar qual a probabilidade de uma classe A ser transferida dado que a classe B foi transferida anteriormente. Foram definidas oito classes na Seção 3.2. Adicionalmente, a matriz de transição ainda possui mais 2 estados: INÍCIO e FIM. Assim, a matriz quadrada a ser definida para cada servidor deve possuir nove linhas e nove colunas.

Foi estabelecido empiricamente que um objeto é o início da requisição quando há um intervalo de no mínimo 20 segundos entre este objeto e o objeto transferido anteriormente. Por consequência, este objeto anterior é considerado o fim da requisição anterior.

Para chegar a este intervalo, foi analisado qual o percentual de requisições que iniciam com a classe HTML para os intervalos de 2, 5, 10, 20 e 120 segundos. É esperado que a maior parte das requisições se inicie em HTML, porém ao se escolher um intervalo pequeno há o risco de considerar uma única requisição como sendo várias, ao passo que um intervalo muito grande pode fazer com que várias requisições sejam consideradas como sendo uma só.

TABELA 4 – ANÁLISE DA CLASSE INICIAL DE REQUISIÇÃO PARA 5 INTERVALOS

| Intervalo | Percentual de requisições que iniciam com HTML |
|---|---|
| 2 segundos | 44% |
| 5 segundos | 47% |
| 10 segundos | 49% |
| 20 segundos | 52% |
| 120 segundos | 56% |

FONTE: O AUTOR (2013)

A diferença do percentual de requisições iniciando em HTML para 20 e 120 segundos não é proporcionalmente tão significativa quando comparada à diferença entre 2 e 20 segundos. Além disso, o intervalo de 20 segundos é condizente com o tempo que se espera que um usuário leia o conteúdo da página e é o tempo utilizado comumente em diversos trabalhos disponíveis na literatura.

TABELA 5 – VARIÁVEIS DA MATRIZ DE PROBABILIDADE DE TRANSIÇÃO

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | $inicio_html | $inicio_gif | $inicio_javascript | $inicio_jpg | $inicio_css | $inicio_png | $inicio_swf | $inicio_outros | $inicio_fim |
| HTML | $html_html | $html_gif | $html_javascript | $html_jpg | $html_css | $html_png | $html_swf | $html_outros | $html_fim |
| GIF | $gif_html | $gif_gif | $gif_javascript | $gif_jpg | $gif_css | $gif_png | $gif_swf | $gif_outros | $gif_fim |
| JPG | $javascript_html | $javascript_gif | $javascript_javascript | $javascript_jpg | $javascript_css | $javascript_png | $javascript_swf | $javascript_outros | $javascript_fim |
| JAVASCRIPT | $jpg_html | $jpg_gif | $jpg_javascript | $jpg_jpg | $jpg_css | $jpg_png | $jpg_swf | $jpg_outros | $jpg_fim |
| CSS | $css_html | $css_gif | $css_javascript | $css_jpg | $css_css | $css_png | $css_swf | $css_outros | $css_fim |
| PNG | $png_html | $png_gif | $png_javascript | $png_jpg | $png_css | $png_png | $png_swf | $png_outros | $png_fim |
| SWF | $swf_html | $swf_gif | $swf_javascript | $swf_jpg | $swf_css | $swf_png | $swf_swf | $swf_outros | $swf_fim |
| OUTROS | $outros_html | $outros_gif | $outros_javascript | $outros_jpg | $outros_css | $outros_png | $outros_swf | $outros_outros | $outros_fim |

FONTE: O AUTOR (2013)

Para realizar esta análise, foi desenvolvido o *script* apresentado no APÊNDICE E. O *script* analisa cada arquivo de sessão de usuário de cada um dos nove servidores, verificando para cada transferência de arquivo qual foi a classe da transferência anterior e por fim incrementa a variável associada a essa transição (Tabela 5). Por exemplo, se o script verificou que na linha 29 de um arquivo de sessão houve uma transferência de um arquivo da classe HTML e na linha 30 houve uma transferência da classe PNG, então a variável “$html_png” é incrementada. Contudo, se o tempo decorrido entre a transferência desses dois arquivos for superior a 20 segundos, o primeiro é considerado o fim de uma requisição e o outro é considerado o início da próxima requisição e portanto as variáveis “$html_fim” e “$inicio_png” são incrementadas.

Após verificar todos os arquivos de sessões de um determinado servidor, o *script* normaliza as variáveis, visto que cada linha da matriz deve ter como soma o valor unitário. Em seguida, o *script* cria um arquivo contendo as probabilidades de cada transição para este servidor. O ciclo é repetido até que seja criada a matriz de transição para todos os servidores.

O fluxograma da Figura 11 apresenta o funcionamento do *script* através de uma simplificação.

As matrizes de probabilidade de transição dos servidores “globoesporte.globo.com” e “www.google.com.br” são exibidas na Tabela 6 e na Tabela 7, respectivamente. As matrizes dos demais servidores são exibidas no APÊNDICE K.

TABELA 6 – MATRIZ DE PROBABILIDADE DE TRANSIÇÃO DO SERVIDOR "GLOBOESPORTE.GLOBO.COM"

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.3846 | 0.0751 | 0.2054 | 0.0697 | 0.2552 | 0 | 0 | 0.0100 | 0 |
| HTML | 0.2384 | 0.0416 | 0.3179 | 0.0240 | 0.2708 | 0 | 0.0009 | 0.0268 | 0.0795 |
| GIF | 0.0233 | 0.5617 | 0.1410 | 0.1813 | 0.0039 | 0 | 0.0012 | 0.0140 | 0.0735 |
| JPG | 0.0197 | 0.1572 | 0.4673 | 0.1019 | 0.1690 | 0 | 0.0002 | 0.0187 | 0.0659 |
| JAVASCRIPT | 0.0082 | 0.1141 | 0.0757 | 0.7467 | 0.0066 | 0 | 0.0003 | 0.0109 | 0.0375 |
| CSS | 0.0163 | 0.1771 | 0.2620 | 0.1077 | 0.3727 | 0 | 0 | 0.0066 | 0.0576 |
| PNG | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| SWF | 0 | 0 | 0.1429 | 0.0476 | 0.0476 | 0 | 0.5714 | 0 | 0.1905 |
| OUTROS | 0.1159 | 0.0429 | 0.1931 | 0.0858 | 0.0987 | 0 | 0 | 0.0300 | 0.4335 |

FONTE: O AUTOR (2013)

TABELA 7 – MATRIZ DE PROBABILIDADE DE TRANSIÇÃO DO SERVIDOR "WWW.GOOGLE.COM.BR"

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.8641 | 0.0540 | 0.0255 | 0.0234 | 0.0002 | 0.0245 | 0.0001 | 0.0082 | 0 |
| HTML | 0.6061 | 0.0366 | 0.0104 | 0.0116 | 0.0004 | 0.0099 | 0 | 0.0042 | 0.3208 |
| GIF | 0.1580 | 0.4596 | 0.0692 | 0.0284 | 0 | 0.1300 | 0 | 0.0594 | 0.0954 |
| JPG | 0.3024 | 0.0754 | 0.3041 | 0.0110 | 0.0018 | 0.0706 | 0 | 0.0644 | 0.1705 |
| JAVASCRIPT | 0.1651 | 0.0723 | 0.0096 | 0.6221 | 0 | 0.0142 | 0 | 0.0046 | 0.1120 |
| CSS | 0.1000 | 0.0333 | 0.3667 | 0 | 0.3667 | 0 | 0 | 0.0667 | 0.0667 |
| PNG | 0.1186 | 0.1177 | 0.0723 | 0.0088 | 0 | 0.6026 | 0 | 0.0298 | 0.0501 |
| SWF | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10000 |
| OUTROS | 0.3193 | 0.0186 | 0.1211 | 0.0059 | 0.0010 | 0.1523 | 0 | 0.1709 | 0.2109 |

FONTE: O AUTOR (2013)

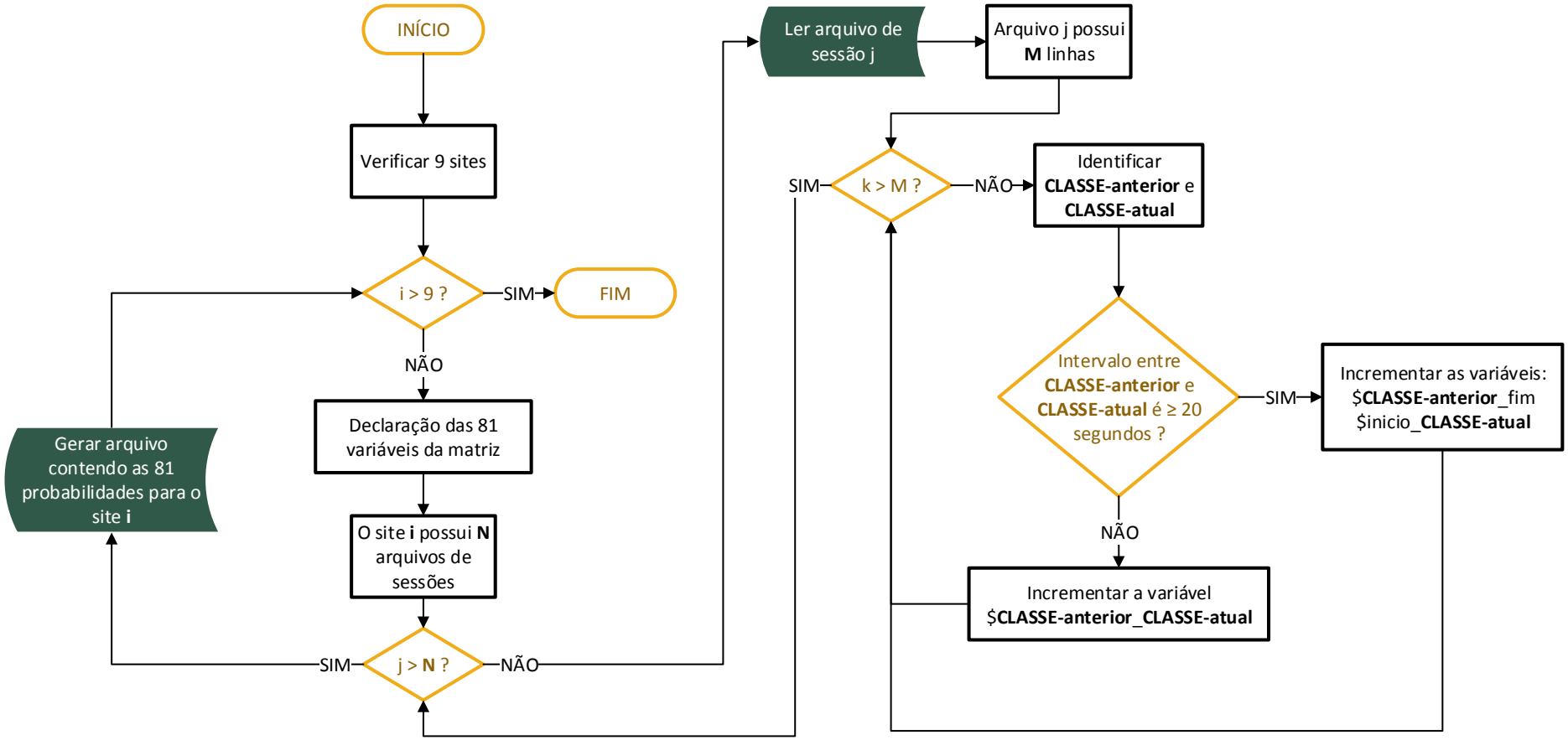



FIGURA 11 – FLUXOGRAMA DO SCRIPT GERADOR DA MATRIZ DE PROBABILIDADE DE TRANSIÇÃO
FONTE: O AUTOR (2013)

### 3.3.1.2. Tamanho de Arquivo de Cada Classe

Cada uma das oito classes do modelo A deve ser modelada separadamente, visto que foi observado em estudos anteriores que a média e desvio padrão varia de acordo com a classe [1].

Inicialmente, é necessário agrupar os tamanhos de cada classe em arquivos separados e, a partir destes dados, realizar uma modelagem estatística. Para isso, foi utilizado *script* apresentado no APÊNDICE F. O *script* lê todos os arquivos de sessões de cada servidor, analisando para cada transferência qual é a classe do arquivo e seu tamanho, em *bytes*. Em seguida, o tamanho é armazenado no arquivo chamado "tamanho_*classe*". Por exemplo, caso o *script* verifique que um arquivo da classe JPG foi transferido e que seu tamanho é de 5900 *bytes*, então o *script* abrirá o arquivo "tamanho_jpg" e adicionará uma nova linha contendo o tamanho (5900).

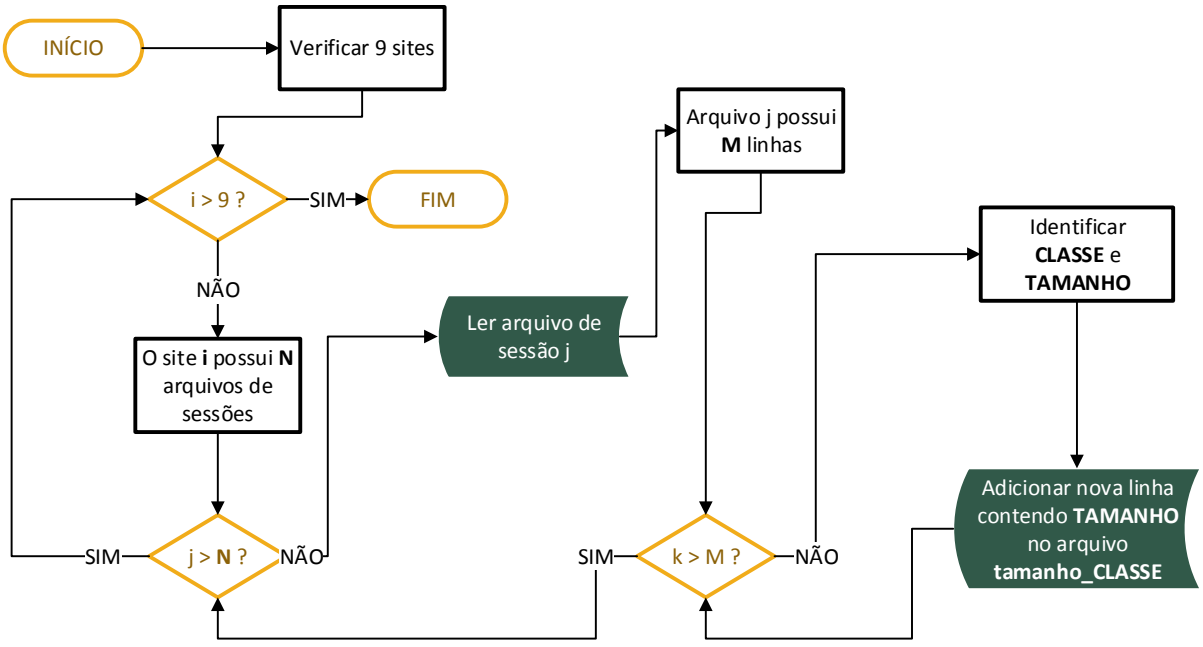



FIGURA 12 – FLUXOGRAMA DO SCRIPT PARA SEPARAR OS TAMANHOS DE CADA CLASSE, DE CADA SERVIDOR, EM UM ARQUIVO POR CLASSE
FONTE: O AUTOR (2013)

Após listar os tamanhos de cada classe em arquivos separados, foi utilizado o *software* R, um ambiente estatístico disponível para sistema operacional Windows, MacOS e plataformas Unix [19].

Para verificar a constatação de estudos anteriores de que o tamanho de arquivos em tráfego *Web* possui cauda pesada, em geral distribuição de Pareto [20, 21], foi analisado o gráfico da distribuição de probabilidade complementar em escala log-log das series contendo tamanho do arquivo de cada classe.

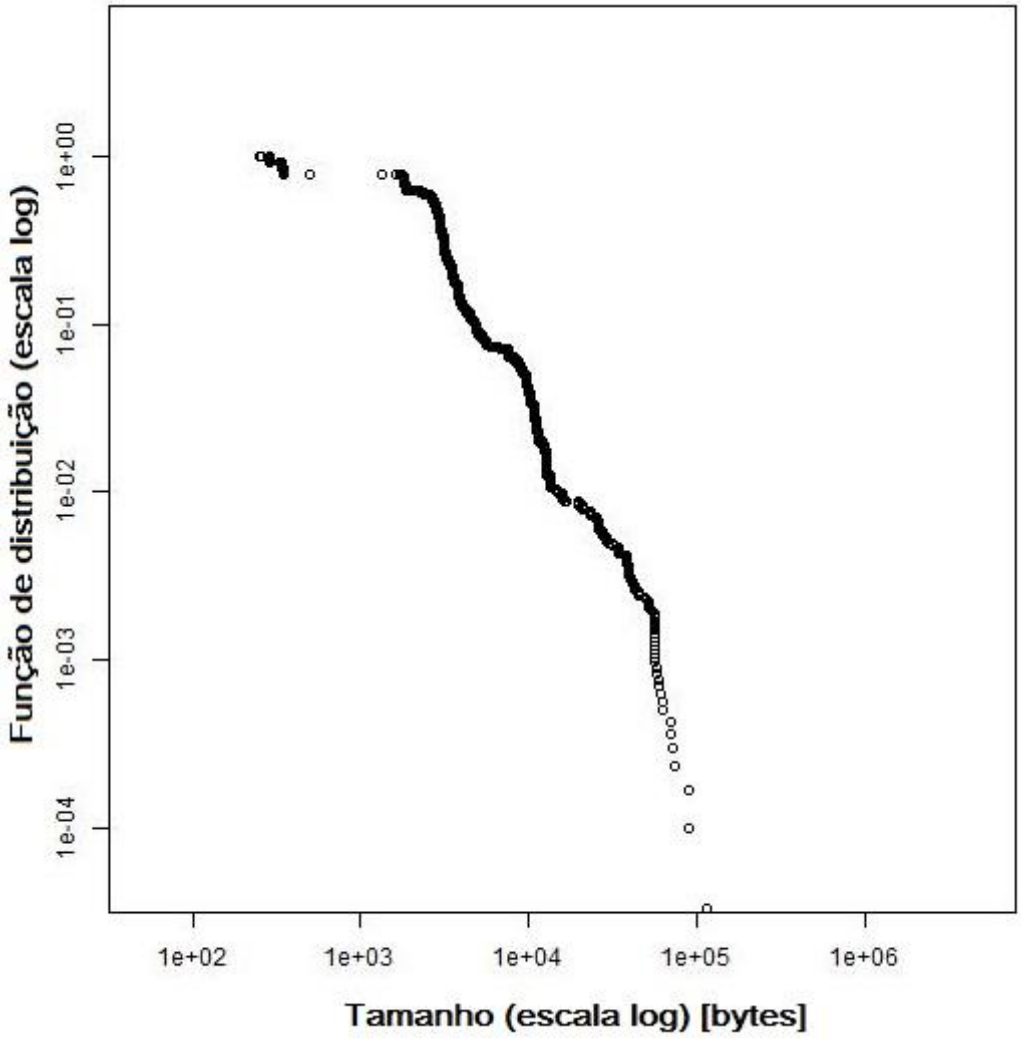



GRÁFICO 1 – FUNÇÃO DE DISTRIBUIÇÃO EM ESCALA LOG-LOG PARA A CLASSE JPG DO SERVIDOR SDP.TERRA.COM.BR
FONTE: O AUTOR (2013)

A função de distribuição em escala log-log da classe JPG do servidor “sdp.terra.com.br”, exibida no Gráfico 1, indica comportamento linear por duas décadas, o que possibilita essa série de valores (tamanho dos arquivos) seja modelada com a distribuição de cauda pesada, como por exemplo a distribuição de Pareto tipo II. O mesmo resultado foi encontrado para as demais séries de todas as classes dos nove servidores.

A distribuição de Pareto possui dois parâmetros: *shape* e *scale*. Para determinar esses parâmetros, o autor utilizou-se do seguinte método:

a) Plotar o gráfico da função de distribuição, em escala log-log;
b) Traçar uma reta sobre a região linear;
c) Medir o ângulo agudo formado entre a reta e o eixo horizontal (Gráfico 2);
d) O parâmetro *shape* será igual à tangente do ângulo medido. Neste exemplo: *shape* = tg (61°) = 1,8;
e) O parâmetro *scale* será igual à média dos valores empíricos multiplicada pelo *shape* menos um. Neste exemplo: *scale* = 3107,9*(1,8 – 1) = 2499;

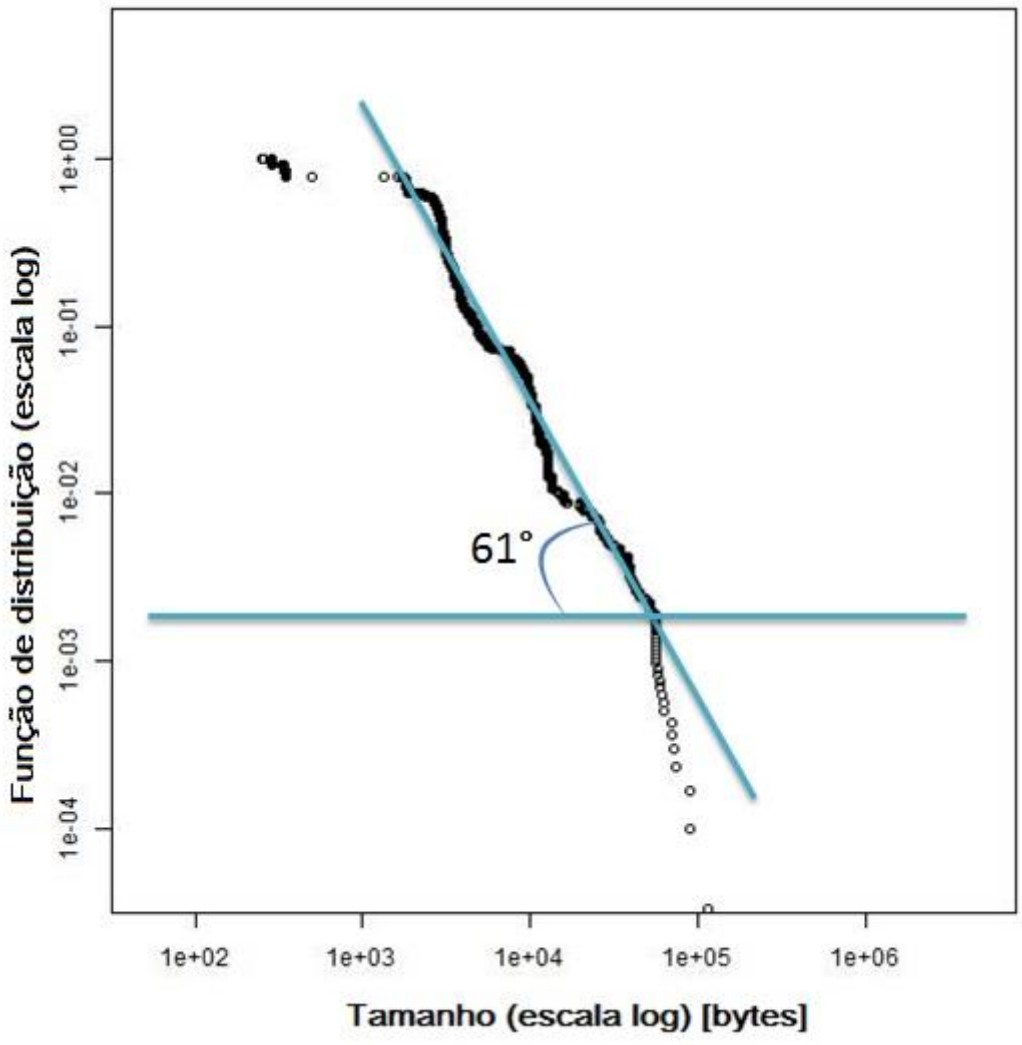



GRÁFICO 2 – MEDIÇÃO DO ÂNGULO FORMADO PELA REGIÃO LINEAR E O EIXO HORIZONTAL
FONTE: O AUTOR (2013)

Com os valores de *shape* e *scale* já definidos, o autor plotou o gráfico Q-Q Plot (quantil-quantil plot). Este gráfico é uma ferramenta estatística utilizada para determinar se dois conjuntos de dados pertencem à mesma distribuição de probabilidade [22].

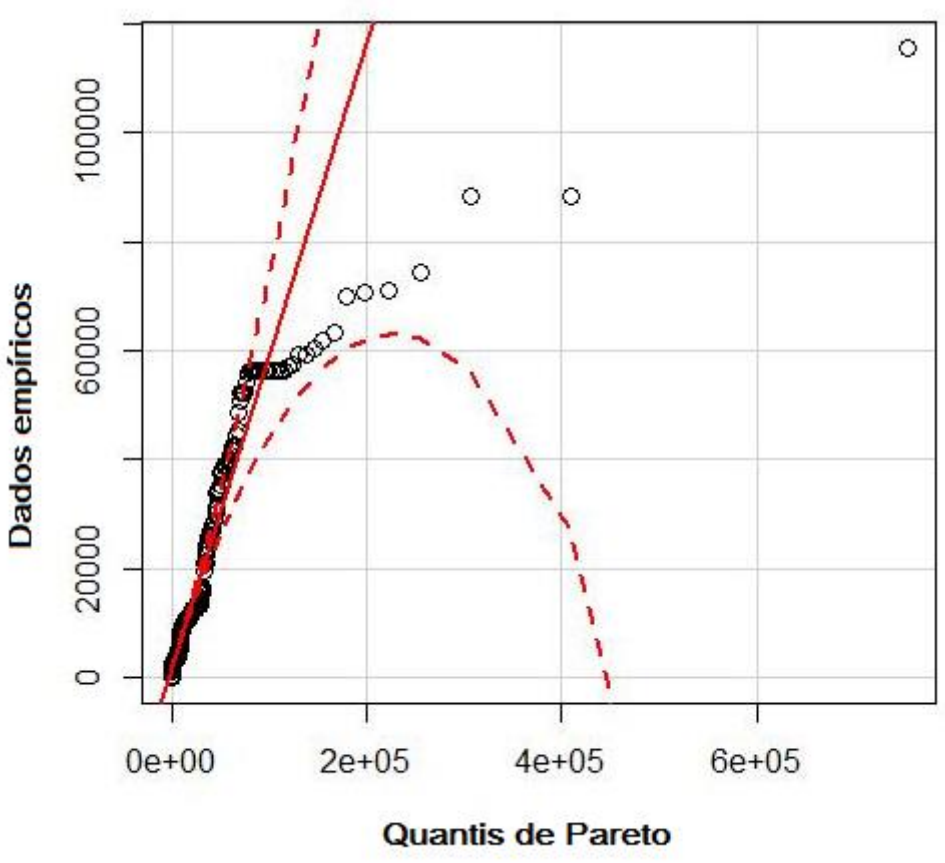



GRÁFICO 3 – Q-Q PLOT DOS DADOS EMPÍRICOS E DISTRIBUIÇÃO DE PARETO COM PARÂMETROS MODELADOS
FONTE: O AUTOR (2013)

Em um gráfico Q-Q Plot, os pontos são formados pelos quantis amostrais, e se eles se aproximam da reta, então as distribuições das duas amostras podem ser consideradas as mesmas. No Gráfico 3 é exibido o Q-Q Plot dos valores de tamanho de arquivos pertencentes à classe JPG, servidor “sdp.terra.com.br”, comparados com quantis da distribuição de Pareto (*shape* = 1,8 e *scale* = 2499). As linhas tracejadas delimitam a região onde o grau de confiança é de 95%.

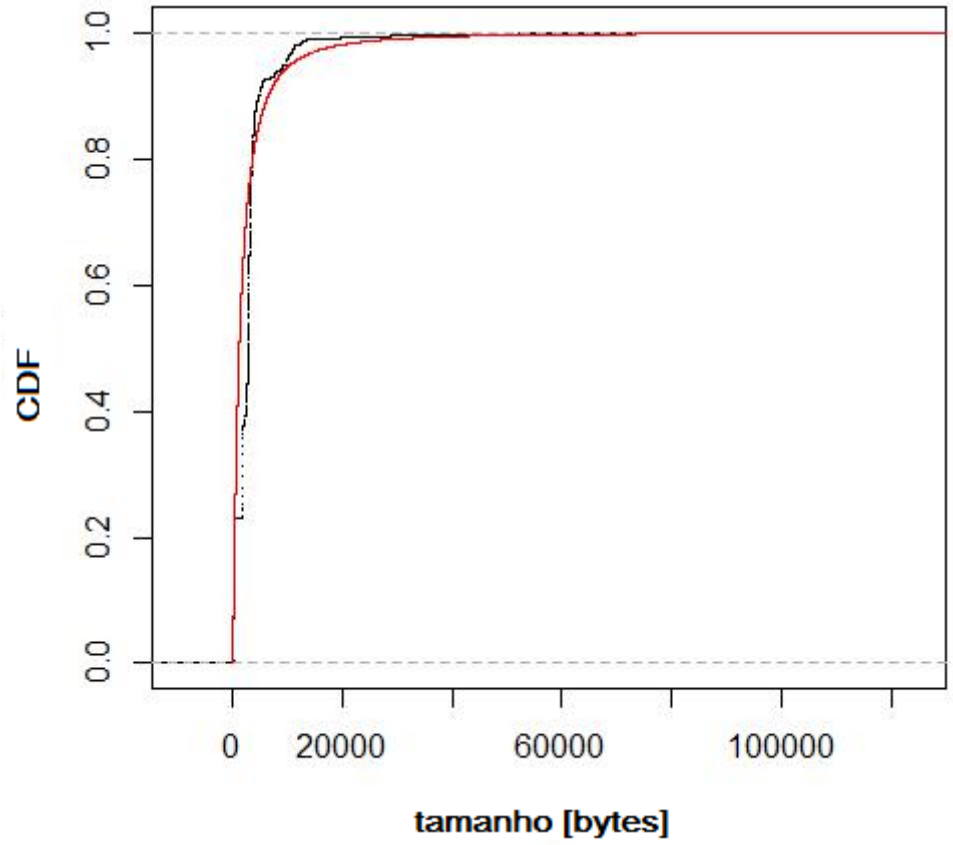



GRÁFICO 4 – CDF DOS VALORES EMPÍRICOS E DA DISTRIBUIÇÃO MODELADA
LINHA PRETA: VALORES EMPÍRICOS; LINHA VERMELHA: DISTRIBUIÇÃO DE PARETO
FONTE: O AUTOR (2013)

O Gráfico 4 mostra a comparação entre a função de distribuição acumulada (CDF) entre os valores de tamanhos empíricos e os valores modelados pela distribuição de Pareto (*shape* = 1,8 e *scale* = 2499).

Na Tabela 8 estão listados os parâmetros definidos para todas as classes dos nove servidores. Todas foram modeladas com a distribuição de Pareto tipo II.

TABELA 8 – VALORES DE *SHAPE* E *SCALE* DA DISTRIBUIÇÃO DE PARETO PARA CADA CLASSE E SERVIDOR

| Servidor | HTML | | GIF | | JPG | | JAVASCRIPT | | CSS | | PNG | | SWF | | OUTROS | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *shape* | *scale* | *shape* | *scale* | *shape* | *scale* | *shape* | *scale* | *shape* | *scale* | *shape* | *scale* | *shape* | *scale* | *shape* | *scale* |
| globoesporte.globo.com | 1,51 | 9996 | 1,24 | 526 | 1,87 | 11191 | 1,51 | 1372 | 1,14 | 1260 | - | - | 1,66 | 20544 | 1,52 | 1122 |
| l.yimg.com | 1,96 | 2514 | 1,62 | 1772 | 1,33 | 2803 | 1,35 | 6948 | 1,08 | 1072 | 1,54 | 3582 | 1,36 | 4588 | 1,25 | 379 |
| mail.google.com | 1,08 | 618 | 1,06 | 169 | 1,22 | 6494 | 1,86 | 17475 | 1,23 | 894 | 1,23 | 1162 | 1,42 | 4835 | 1,28 | 16581 |
| sdp.terra.com.br | 1,78 | 2580 | 1,26 | 436 | 1,8 | 2499 | - | - | 1,22 | 319 | 1,99 | 3017 | - | - | 1,15 | 43 |
| stf.terra.com.br | 1,56 | 715 | 1,72 | 1247 | 1,53 | 3937 | 1,21 | 623 | 1,63 | 1547 | 1,44 | 374 | 1,68 | 4926 | 1,07 | 7392 |
| www.globo.com | 1,04 | 749 | 1,3 | 295 | 1,83 | 3357 | 1,23 | 639 | 1,66 | 1230 | 1,15 | 504 | - | - | 1,53 | 582 |
| www.google.com | 1,31 | 881 | 1,96 | 1687 | 1,99 | 2514 | 1,17 | 2018 | 1,07 | 581 | 1,46 | 2743 | - | - | 1,89 | 1722 |
| www.google.com.br | 1,99 | 3599 | 1,26 | 850 | 2 | 2862 | 1,93 | 5939 | 1,28 | 1636 | 1,96 | 2528 | 1,49 | 188 | 1,95 | 1722 |
| www.icisaude.org.br | 1,27 | 332 | 1,36 | 213 | 1,08 | 397 | 1,26 | 927 | 1,99 | 416 | 1,12 | 133 | - | - | 1,13 | 337 |

FONTE: O AUTOR (2013)

### 3.3.2 Parâmetros do Modelo B

O modelo B representa as requisições como sendo um conjunto do objeto principal, seguido de objetos embutidos, não os diferenciando por classe, e possuindo uma distribuição de probabilidade para cada uma das três categorias: tamanho dos objetos principal e embutidos, e número de objetos embutidos em cada requisição.

O objeto principal é o primeiro objeto transferido em cada requisição. Na Seção 3.3.1.1 o autor convenciona que uma requisição começa quando o intervalo entre uma transferência de arquivo e a transferência anterior é igual ou superior a 20 segundos. Na parametrização do modelo B esta convenção também foi utilizada.

Para determinar quais distribuições podem representar os tamanhos dos objetos principal e embutido, e a quantidade de objetos embutidos em cada requisição, foram realizadas etapas semelhantes às descritas na Seção 3.3.1.2. Primeiramente foi necessário separar atributos em arquivos separados para fazer a análise estatística.

O *script* do APÊNDICE G faz a leitura de cada sessão de cada um dos nove servidores, verifica para cada requisição qual o tamanho da primeira transferência e insere este valor em uma nova linha no arquivo “main_object_size” (tamanho do objeto principal). O mesmo é feito com os tamanhos dos objetos embutidos, gravados no arquivo “inline_object_size”, e com quantidades de objetos embutidos por requisição, salvos no arquivo “inline_object_number”.

O fluxograma da Figura 13 ilustra o funcionamento deste *script* simplificadamente.

Fazendo a análise da função de distribuição em escala log-log – procedimento descrito na Seção 3.3.1.2 – foi observado que os parâmetros de tamanho dos objetos principal e embutidos apresentavam comportamento semelhantes a quantis de Pareto.

Para a quantidade de objetos embutidos, a distribuição que melhor se encaixou foi a Exponencial. Esta distribuição tem como único parâmetro o *rate*, que pode ser calculado pelo inverso da média dos dados empíricos.

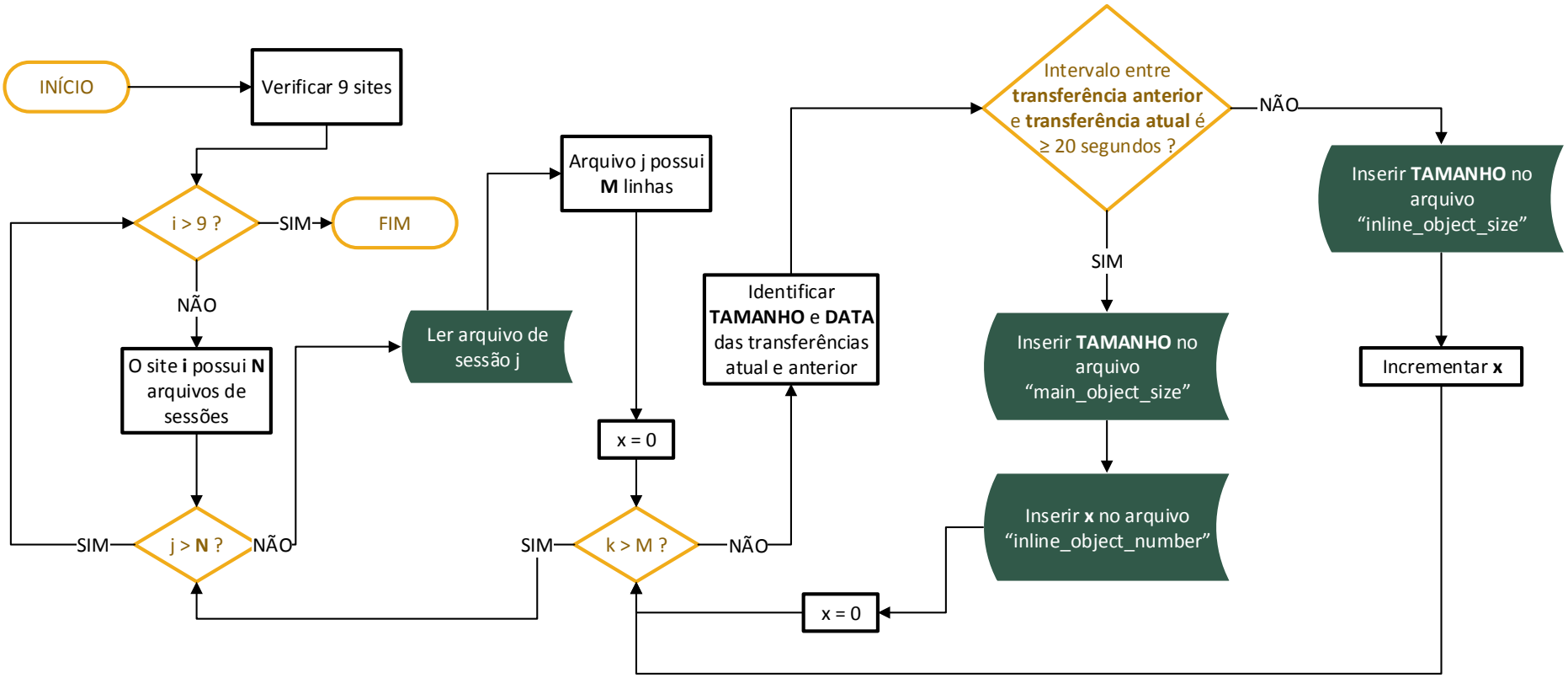



FIGURA 13 – FLUXOGRAMA DO SCRIPT UTILIZADO PARA SEPARAR ATRIBUTOS DO MODELO B EM ARQUIVOS SEPARADOS
FONTE: O AUTOR (2013)

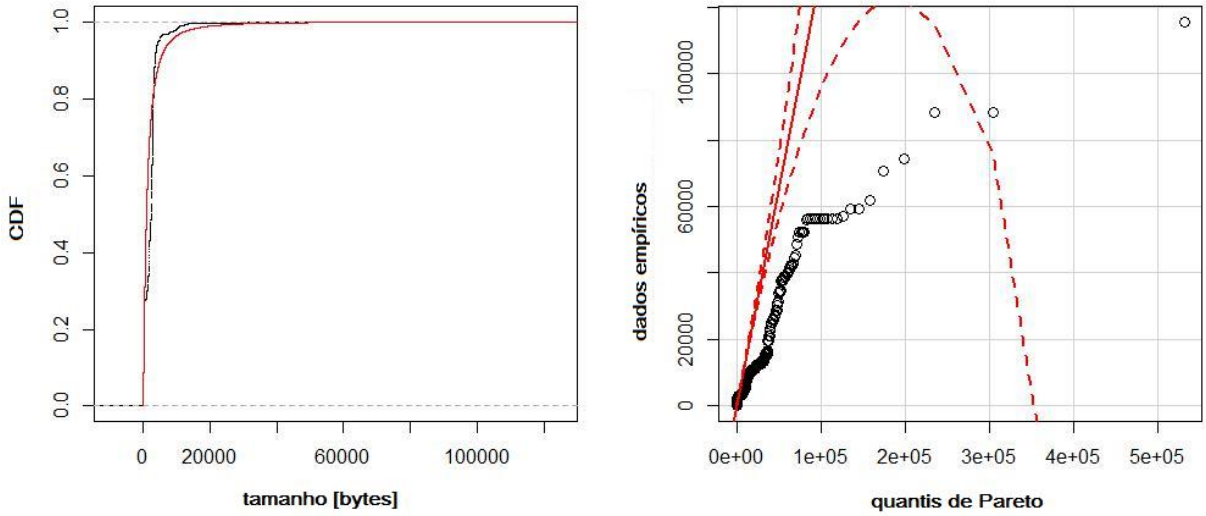



GRÁFICO 5 – TAMANHO DO OBJETO PRINCIPAL DO SERVIDOR “SDP.TERRA.COM.BR”
DISTRIBUIÇÃO DE PARETO (*SHAPE*=1,7; *SCALE*=2452)
À ESQUERDA: CDF (LINHA VERMELHA: VALORES TEÓRICOS)
À DIREITA: Q-Q PLOT
FONTE: O AUTOR (2013)

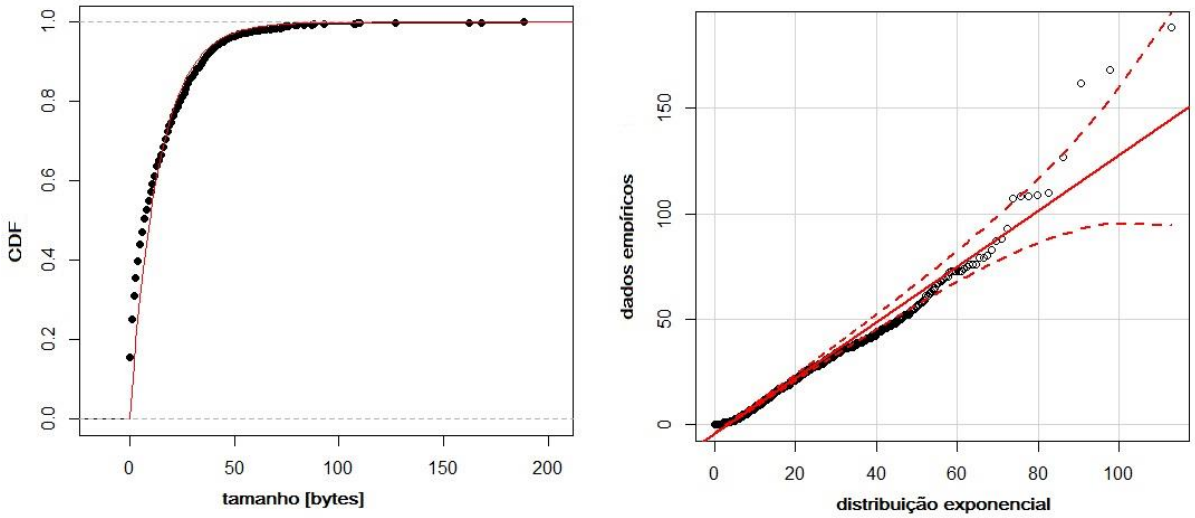



GRÁFICO 6 – QUANTIDADE DE OBJETOS EMBUTIDOS DO SERVIDOR “SDP.TERRA.COM.BR”
DISTRIBUIÇÃO EXPONENCIAL (*RATE*=0,07)
À ESQUERDA: CDF (LINHA VERMELHA: VALORES TEÓRICOS)
À DIREITA: Q-Q PLOT
FONTE: O AUTOR (2013)

Os parâmetros do modelo B encontrados para cada servidor estão listados na Tabela 9.

TABELA 9 – PARÂMETROS DO MODELO B PARA CADA SERVIDOR

| Servidor | Tamanho do objeto principal (Distribuição de Pareto) | | Tamanho do objeto embutido (Distribuição de Pareto) | | Quantidade de objetos embutidos por requisição (Distribuição Exponencial) |
|---|---|---|---|---|---|
| | *shape* | *scale* | *shape* | *scale* | *rate* |
| globoesporte.globo.com | 1,75 | 7070 | 1,46 | 3620 | 0,067 |
| l.yimg.com | 1,59 | 3298 | 1,15 | 1305 | 0,112 |
| mail.google.com | 1,03 | 725 | 1,06 | 854 | 0,169 |
| sdp.terra.com.br | 1,66 | 2452 | 1,98 | 2318 | 0,072 |
| stf.terra.com.br | 1,42 | 1042 | 1,35 | 1150 | 0,069 |
| www.globo.com | 1,23 | 887 | 1,56 | 2043 | 0,084 |
| www.google.com | 1,49 | 1840 | 1,41 | 2048 | 0,318 |
| www.google.com.br | 1,98 | 3769 | 1,99 | 3464 | 0,339 |
| www.icisaude.org.br | 1,99 | 4963 | 1,34 | 597 | 0,099 |

FONTE: O AUTOR (2013)

### 3.3.3 Parâmetros em Comum

Alguns parâmetros são comuns entre os modelos A e B, devido ao fato de ambos terem base no modelo SURGE. São eles:

- Tempo de OFF: para ambos os modelos, esse tempo foi definido como sendo, no mínimo, 20 segundos. Isto se deu pelo fato de que o tempo de OFF é o intervalo entre as requisições, onde em geral o usuário está lendo o conteúdo; apesar de se ter convencionado o limite inferior para este parâmetro, também é necessário encontrar uma distribuição de probabilidade que represente-o;
- Tempo de *active-off*: para o modelo A, este parâmetro representa o período entre cada transição do diagrama de estados (Figura 5), enquanto que no modelo B é o intervalo de tempo entre a transferência

de cada objeto da mesma requisição – objeto principal e primeiro objeto embutido, e entre cada objeto embutido;

- Quantidade de requisições por sessão: outro parâmetro que representa o comportamento do usuário é o número de requisições que são feitas em uma mesma sessão; este parâmetro também foi necessário para realizar as simulações de ambos os modelos;

O *script* do APÊNDICE E, ao mesmo tempo em que gera a matriz de probabilidade de transição para o modelo A, coleta os valores dos três parâmetros acima e separa-os em três arquivos para que seja feita a análise estatística.

Em estudos anteriores [1, 2, 7] foi observado que os tempos de OFF e *active-off* em geral seguem distribuição de Weibull. Com o resultado da análise estatística dos valores separados pelo *script*, esta aderência foi confirmada. Utilizando a ferramenta “fitdist”, da biblioteca “fitdistrplus” do ambiente R, o autor ajustou as series empíricas de intervalos de OFF e *active-off* em distribuição de Weibull. Já as quantidades de requisições por sessão de usuário seguiram distribuição Exponencial.

No Gráfico 7 são comparados os valores empíricos e distribuição de Weibull, para intervalo de OFF observado nos acessos ao servidor “www.google.com”.

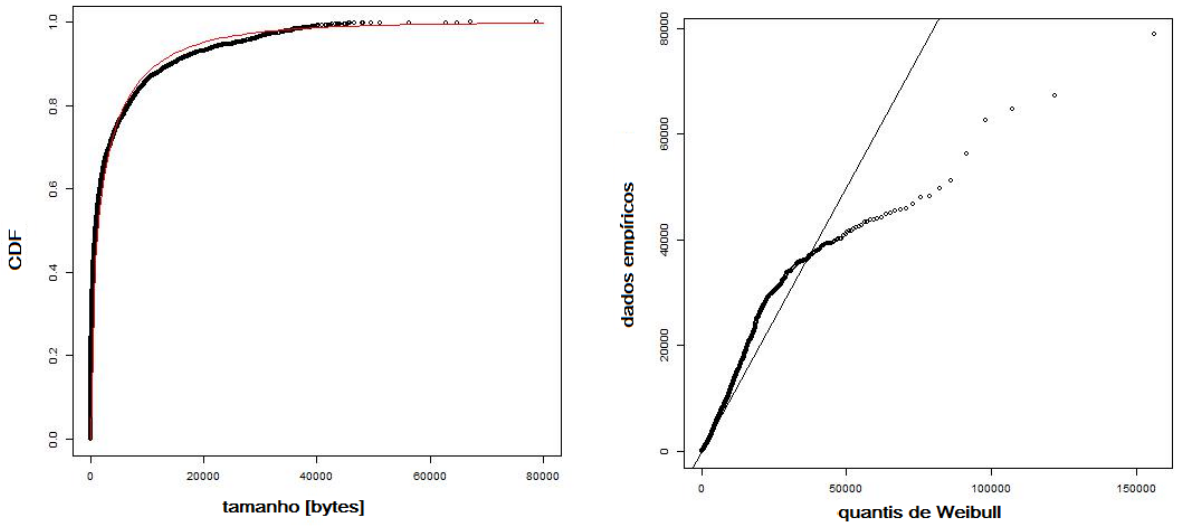



GRÁFICO 7 – TEMPO DE OFF DO SERVIDOR “WWW.GOOGLE.COM”
DISTRIBUIÇÃO DE WEIBULL (*SHAPE*=0,52; *SCALE*=2432)
À ESQUERDA: CDF (LINHA VERMELHA: VALORES TEÓRICOS)
À DIREITA: Q-Q PLOT

Os parâmetros encontrados para tempos de OFF e *active-off*, e quantidade de requisições por sessão de usuários que foram usados para os modelos A e B são listados na Tabela 10.

TABELA 10 – PARÂMETROS DE TEMPO OFF, *ACTIVE-OFF* E QUANTIDADE DE REQUISIÇÕES

| Servidor | Tempo OFF (Distribuição de Weibull) | | Tempo *active-off* (Distribuição de Weibull) | | Quantidade de requisições por sessão de usuário (Distribuição Exponencial) |
|---|---|---|---|---|---|
| | *shape* | *scale* | *shape* | *scale* | *rate* |
| globoesporte.globo.com | 0,46 | 572 | 0,48 | 0,19 | 0,081 |
| l.yimg.com | 0,45 | 1034 | 0,45 | 0,16 | 0,126 |
| mail.google.com | 0,53 | 606 | 0,48 | 0,61 | 0,168 |
| sdp.terra.com.br | 0,48 | 489 | 0,44 | 0,12 | 0,082 |
| stf.terra.com.br | 0,47 | 524 | 0,46 | 0,17 | 0,077 |
| www.globo.com | 0,49 | 816 | 0,48 | 0,14 | 0,089 |
| www.google.com | 0,52 | 2432 | 0,46 | 0,41 | 0,301 |
| www.google.com.br | 0,51 | 1123 | 0,45 | 0,59 | 0,282 |
| www.icisaude.org.br | 0,58 | 293 | 0,41 | 0,31 | 0,092 |

FONTE: O AUTOR (2013)

## 3.4 Simulações

### 3.4.1 Método Para Geração De Valores Aleatórios

Na Seção 3.3 foram apresentadas as distribuições de probabilidade utilizadas para caracterizar as diversas variáveis do modelo. No momento das simulações, é necessário gerar valores aleatórios para cada variável de modo que pertençam às mesmas distribuições de probabilidade com os mesmos parâmetros observados empiricamente.

O método utilizado pelo autor consiste em isolar a variável aleatória da função de distribuição acumulada P(X ≤ x), posteriormente substituindo P(X ≤ x) por um número aleatório fracionário entre 0 e 1.

Por exemplo, para a distribuição de Pareto tipo II, usada várias vezes neste trabalho, temos a seguinte CDF:

$$\mathrm{F(x) = P(X \leq x) = \left(1 + \frac{x}{b}\right)^{-a}}, a > 1$$

Onde $a$ é o *shape* e $b$ é *scale*. Isolando $x$, temos:

$$x = -b\left(1 - F(x)^{-\frac{1}{a}}\right)$$

Conhecendo os parâmetros $a$ e $b$, e substituindo F(x) por um número aleatório, chega-se ao valor desejado $x$.

Para gerar o número aleatório fracionário entre 0 e 1, o autor utilizou a função *rand()* presente na linguagem de programação Perl.

### 3.4.2 Simulação Do Modelo A

Para realizar simulações do modelo A, foi desenvolvido o *script* do APÊNDICE H, que simula várias sessões de usuário e salva o *log* de cada sessão em um arquivo de texto separado, de modo que esses dados possam ser analisados posteriormente. Cada transferência é representada em uma linha, no seguinte formato:

*<Tempo em que a transferência foi realizada, em segundos> <Tamanho do arquivo transferido, em bytes>*

Para gerar uma sessão de usuários simulada pelo modelo A, primeiramente deve-se definir quantas requisições serão realizadas nesta sessão. Esta quantidade

é definida por uma variável aleatória com distribuição Exponencial, conforme observado na Seção 3.3.3.

Em sequencia, inicia-se a primeira requisição, de modo que um intervalo de OFF é gerado, com distribuição de Weibull, e a primeira transferência da requisição é feita nesse tempo. Para definir qual é a classe de arquivo da primeira transferência, um número aleatório fracionário entre 0 e 1 é gerado através da função Perl *rand()* e esse número, então, é comparado com os valores acumulados dos elementos da linha INÍCIO da matriz de probabilidade de transição.

Por exemplo, o servidor “www.icisaude.org.br” possui a matriz da Tabela 6, cuja linha INÍCIO é:

TABELA 11 – LINHA INÍCIO DA MATRIZ DE PROBABILIDADE DE TRANSIÇÃO DO SERVIDOR “WWW.ICISAUDE.ORG.BR”

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.0174 | 0.1136 | 0.3366 | 0 | 0.1792 | 0.0174 | 0 | 0.3360 | 0 |

FONTE: O AUTOR (2013)

Os valores acumulados dessa linha INÍCIO, portanto, são:

TABELA 12 – LINHA INÍCIO COM VALORES ACUMULADOS

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.0174 | 0.1310 | 0.4676 | 0 | 0.6468 | 0.6642 | 0 | 1.000 | 0 |

FONTE: O AUTOR (2013)

Se, por exemplo, o número aleatório gerado for 0,0105 então a primeira transferência será da classe HTML, uma vez que:

$$0 < 0{,}0105 \leq 0{,}0174$$

Essa dinâmica é utilizada não apenas para definir a classe inicial da requisição, mas em todos os momentos em que houver transição de classes.

Após definida a classe, o tamanho do arquivo, que seguirá distribuição de Pareto tipo II (Seção 3.3.1.2), é gerado com os parâmetros de *scale* e *shape* referentes àquela classe.

Uma vez que o tempo e o tamanho do arquivo foram definidos, uma linha é adicionada ao arquivo de sessão, contendo esses valores e assim a transferência inicial é encerrada.

Para iniciar a próxima transferência da requisição, um novo número aleatório fracionário é gerado e comparado com a matriz de probabilidade de transição. Todavia, neste momento duas situações podem ocorrer: o número aleatório pode indicar transição a transição de um novo arquivo (podendo ser, ou não, idêntica à classe transferida anteriormente) ou pode indicar o fim da requisição.

Caso a próxima transição seja a transferência de um arquivo, um intervalo de *active-off* é gerado (Distribuição de Weibull) e o tempo em que este novo arquivo será transferido é o dado pelo tempo em que o arquivo anterior foi transferido somado deste intervalo. Porém, caso a próxima transição indique o fim da requisição, então um tempo de OFF é gerado e somado ao tempo da transferência anterior para representar o tempo da primeira transferência da próxima requisição.

O ciclo se repete até que seja gerada a quantidade de requisições por sessão definida previamente.

A Figura 14 mostra um fluxograma que representa como o *script* simula as sessões usando o modelo A.

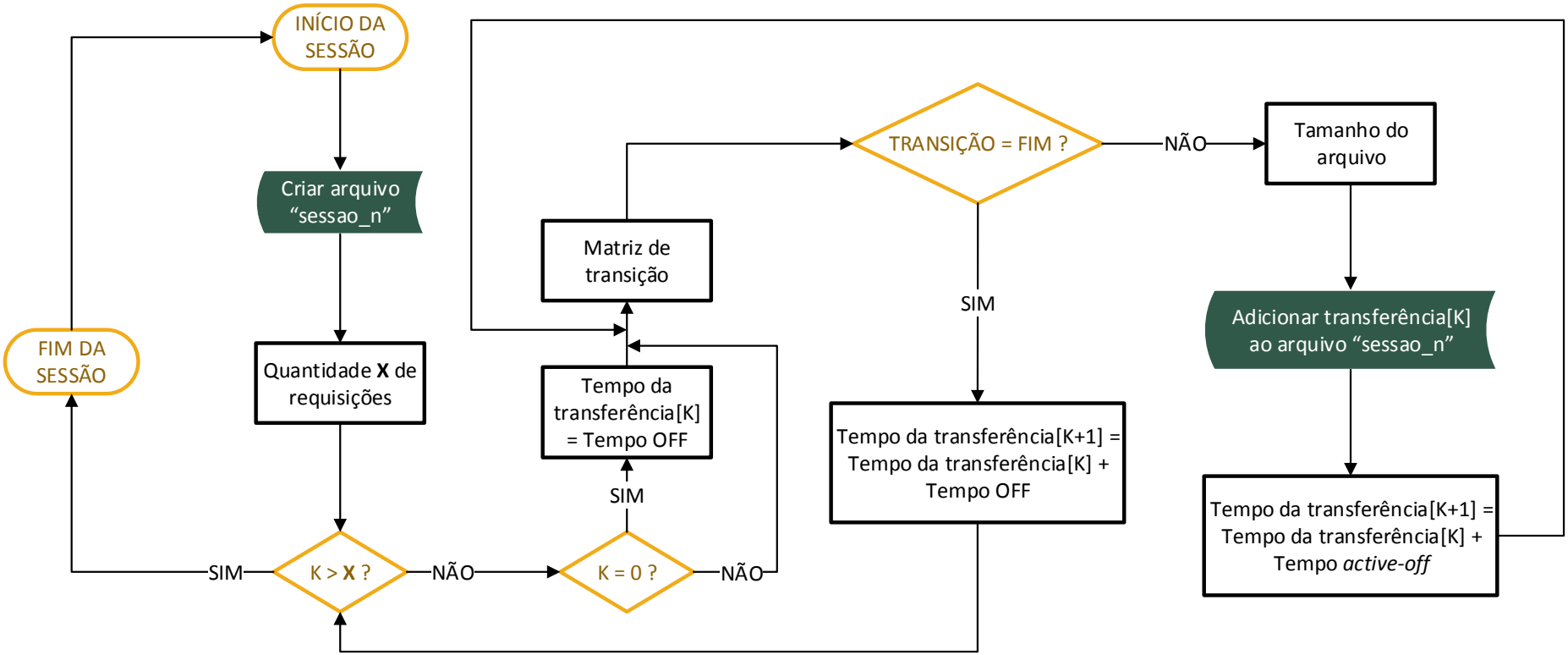



FIGURA 14 – FLUXOGRAMA DE SIMULAÇÃO DE SESSÃO COM MODELO A
FONTE: O AUTOR (2013)

### 3.4.3 Simulação Do Modelo B

A geração de sessões de usuários a partir do modelo B foi feita com o *script* do APÊNDICE I. Assim como nas simulações do modelo A, cada sessão simulada é registrada em um arquivo de texto único, de modo que cada linha deste arquivo representa a transferência de um arquivo.

No inicio da simulação, é determinada a quantidade de requisições que serão feitas na sessão. Em seguida, inicia-se a primeira requisição, de modo que o primeiro arquivo transferido nessa requisição é o objeto principal. O tempo em que esse arquivo é transferido é determinado pelo intervalo de OFF (distribuição de Weibull), e o tamanho do arquivo (objeto principal) é determinado com base na distribuição de Pareto tipo II.

Após a transferência do objeto principal, calcula-se a quantidade de objetos embutidos com a distribuição Exponencial. Uma vez que esta é uma distribuição contínua, é necessário arredondar esse valor para o inteiro mais próximo. Assim, transfere-se cada objeto embutido com intervalo de *active-off*, sendo que depois último objeto embutido é calculado um intervalo de OFF para iniciar a próxima requisição.

A Figura 15 mostra o fluxograma de como o *script* simula sessões de usuário a partir do modelo B.

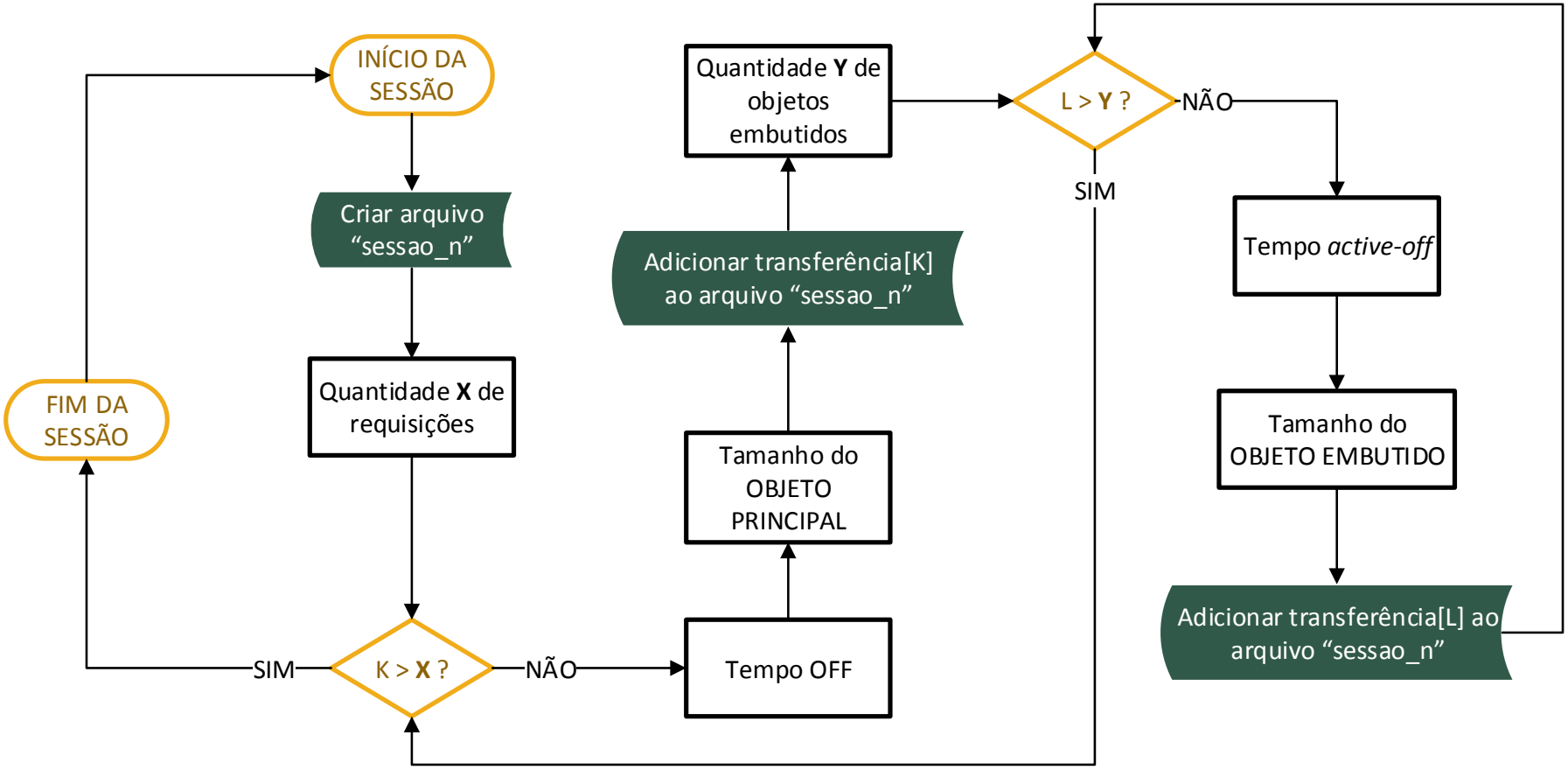



FIGURA 15 – FLUXOGRAMA DE SIMULAÇÃO DE SESSÃO COM MODELO B
FONTE: O AUTOR (2013)

# 4 Análise Dos Resultados

A comparação realizada neste trabalho entre modelo A, modelo B e tráfego real foi feita com base nos níveis de autocorrelação dos tamanhos de arquivos transferidos, visto que essa característica é decisiva para se dimensionar capacidades de transmissão, *buffers* e prever o desempenho de servidores [1]. Neste sentido, um modelo preciso é aquele que produz nível de autocorrelação próximo ao produzido pelo tráfego real.

Neste trabalho, a análise de autocorrelação foi feita para cada um dos nove servidores, comparando os dois modelos e o tráfego real (*trace*), em dois níveis:

a) Autocorrelação no tamanho dos arquivos em uma única sessão
b) Autocorrelação no tamanho dos arquivos em múltiplas sessões simultâneas

A quantidade de sessões simuladas usando cada um dos modelos para cada servidor é o número de sessões de usuário que aquele servidor obteve no *trace* da PUCPR. Para se fazer a análise de múltiplas sessões simultâneas, o autor agrupou todas as sessões simuladas para o servidor e as ordenou cronologicamente.

TABELA 13 – QUANTIDADE DE SESSÕES POR SERVIDOR

| Servidor | Quantidade de sessões |
|---|---|
| globoesporte.globo.com | 374 |
| l.yimg.com | 909 |
| mail.google.com | 943 |
| sdp.terra.com.br | 413 |
| stf.terra.com.br | 550 |
| www.globo.com | 847 |
| www.google.com | 1647 |
| www.google.com.br | 2215 |
| www.icisaude.org.br | 81 |

FONTE: O AUTOR (2013)

A autocorrelação foi medida através da soma dos tamanhos de arquivos transferidos em intervalos de 1 segundo. Este tempo está relacionado com tamanho de *buffers* de servidores. O *script* usado para fazer esta soma a cada segundo e criar séries para serem análisadas é mostrado no APÊNDICE J.

As séries de somatório de tamanho de arquivos foram então inseridas na função de autocorrelação (ACF) do *software* R.

Os níveis de autocorrelação encontrados foram similares para os modelos A e B e tráfego real quando a análise foi feita no nível de sessão única, indicando que se tratando de sessões individuais ambos os modelos geram resultados similares ao tráfego real.

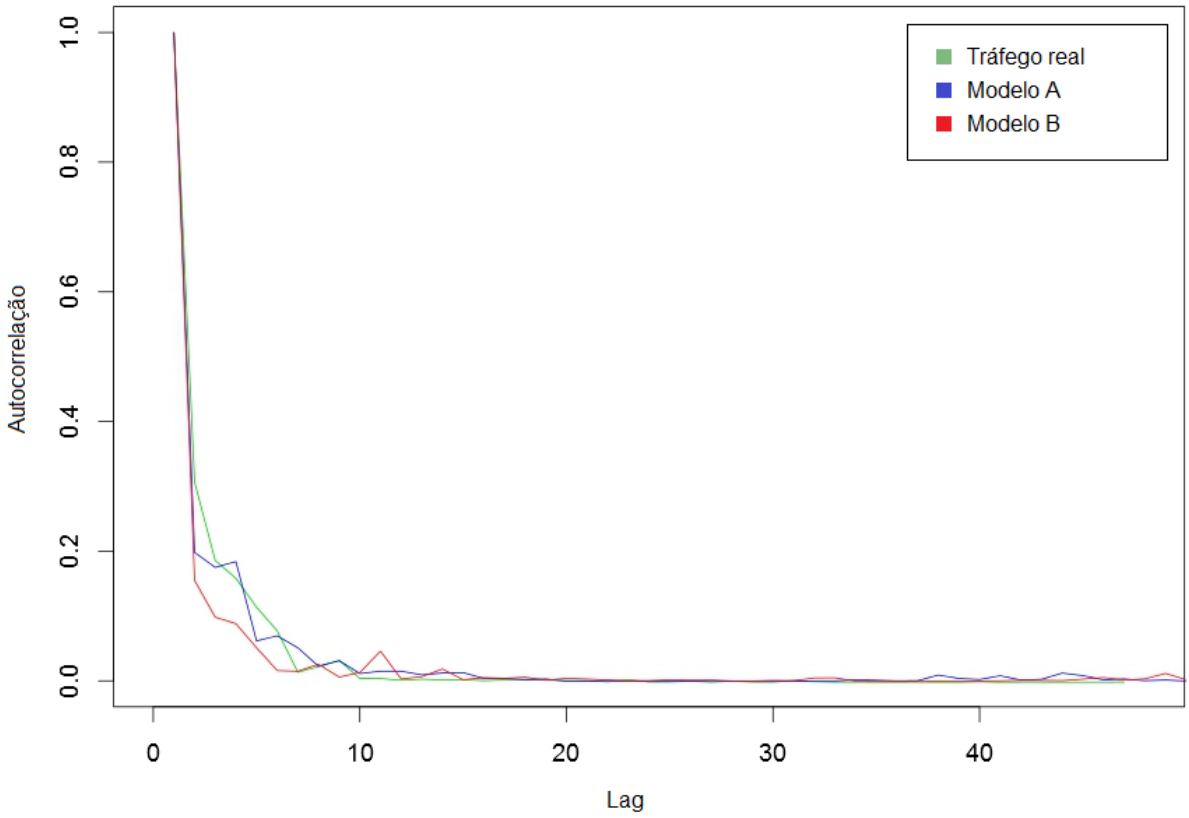



GRÁFICO 8 – AUTOCORRELAÇÃO DE SESSÃO ÚNICA DO SERVIDOR “WWW.GLOBO.COM”
FONTE: O AUTOR (2013)

Contudo, na análise em nível de múltiplas sessões simultâneas, que é o comportamento real observado em servidores (cada servidor é acessado por múltiplos usuários), observou-se que em geral o modelo B apresentou níveis de autocorrelação maiores do que o tráfego real, enquanto que o modelo A se manteve similar ao caso real (Gráfico 9). Neste caso, o modelo A se sobressai em relação ao modelo B, podendo ser considerado mais representativo ao tráfego real ao que se diz respeito da autocorrelação encontrada no tamanho dos arquivos transmitidos.

Este comportamento é observado no Gráfico 9 e Gráfico 10. Os gráficos de ACF dos demais servidores são exibidos no APÊNDICE L.

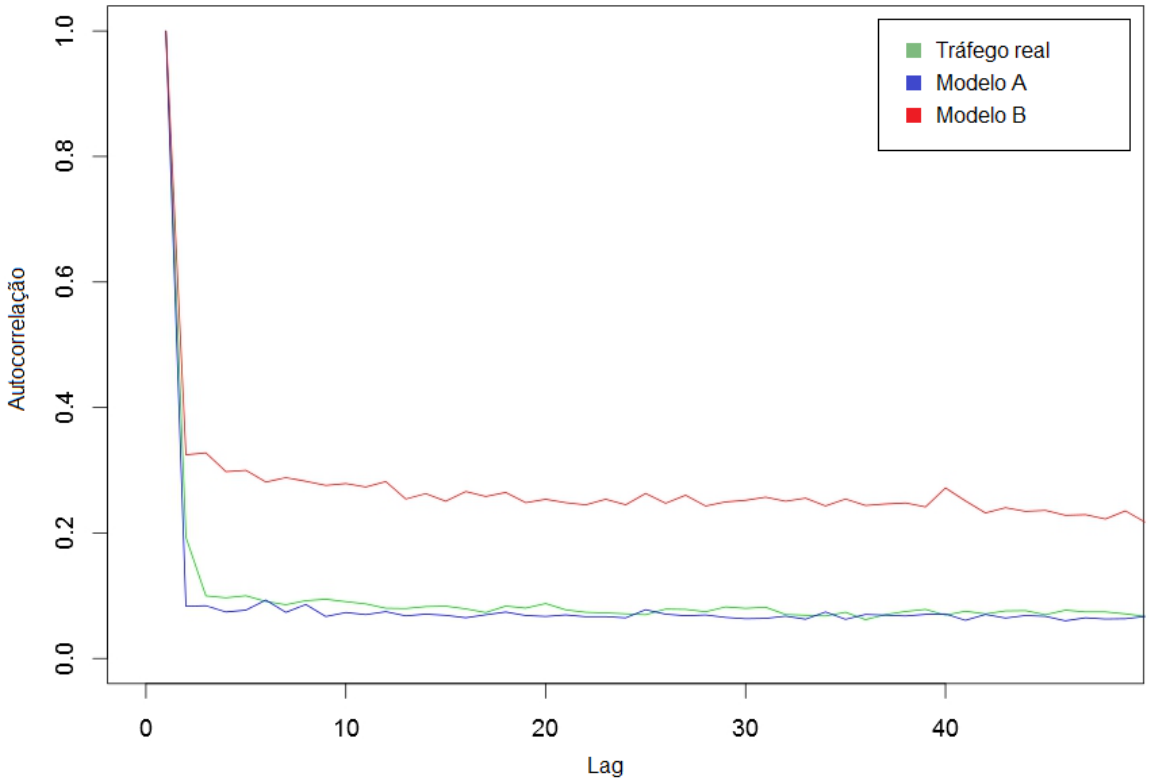



GRÁFICO 9 – AUTOCORRELAÇÃO DE MÚLTIPLAS SESSÕES DO SERVIDOR "WWW.GOOGLE.COM.BR"
FONTE: O AUTOR (2013)

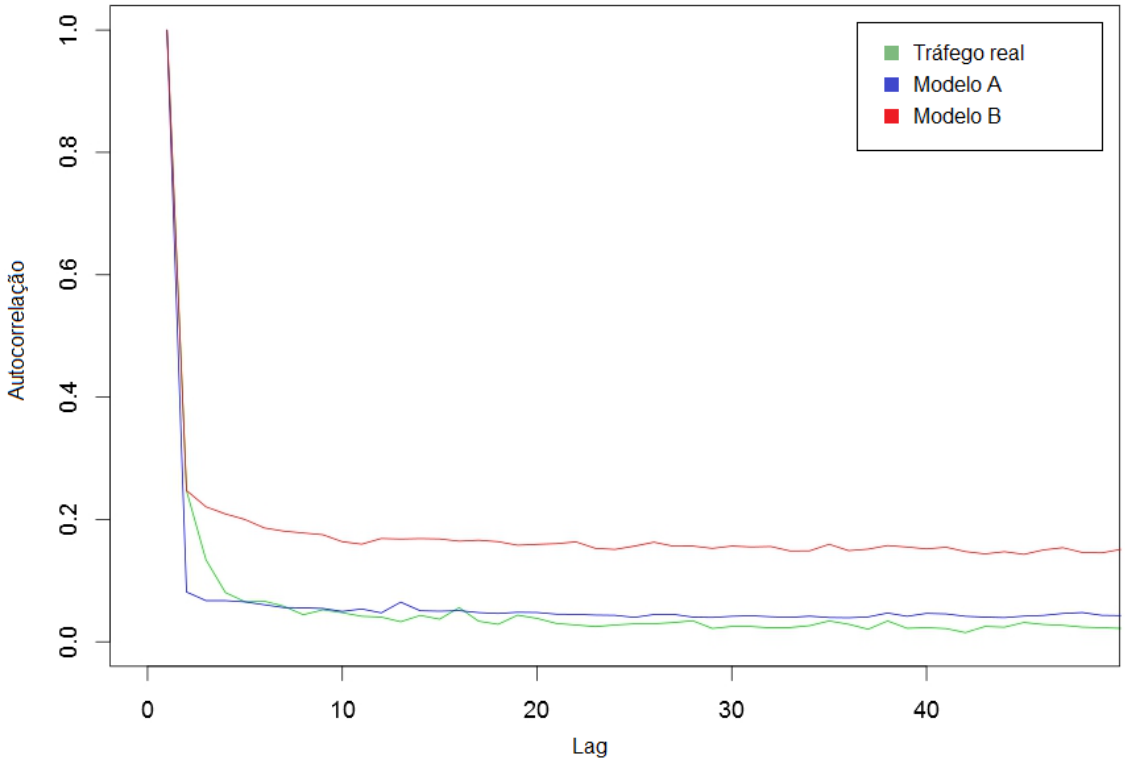



GRÁFICO 10 – AUTOCORRELAÇÃO DE MÚLTIPLAS SESSÕES DO SERVIDOR "SDP.TERRA.COM.BR"

FONTE: O AUTOR (2013)

## 5 Conclusão

Esta monografia apresentou a análise comparativa entre níveis de autocorrelação nos tamanhos de arquivos transmitidos em uma amostra de tráfego real e em simulações utilizando os modelos de representação de tráfego *Web* propostos por Pedroso e Fonseca e por Choi e Limb.

Para se chegar aos resultados, o autor necessitou inicialmente realizar o processamento de dados do *log* coletado no servidor *proxy* da PUCPR, parametrizar os dois modelos em teste e gerar simulações de sessões de usuários com cada um dos modelos. Estas etapas foram cumpridas a partir do desenvolvimento de *scripts* em linguagem Perl e análise estatística com o *software* R.

A caracterização do tráfego de um servidor *Web* através do modelo proposto por Pedroso e Fonseca requer as seguintes informações:

- Classes de arquivos transmitidos no servidor
- Distribuições de probabilidade e parâmetros para representação do tamanho de arquivo transmitido para cada classe
- Probabilidades de transição de estados para transmissão de classes de arquivos em uma requisição
- Quantidade de requisições por sessão
- Intervalos de OFF e *active-off*

O modelo proposto por Pedroso e Fonseca mostrou maior similaridade ao tráfego real do que o modelo de Choi e Limb, no que diz respeito à autocorrelação do tamanho de arquivos transmitidos em simulações com múltiplas sessões de usuários. Esse é um importante aspecto para o dimensionamento de servidores e previsão de desempenho.

Como trabalho futuro, pode-se implementar o modelo proposto por Pedroso e Fonseca no *software* NS-2, um simulador de rede amplamente utilizado pela comunidade acadêmica, a fim de verificar os diversos comportamentos e características do modelo em cenários mais complexos.

# REFERÊNCIAS

[1] PEDROSO, C. M.; FONSECA, K. **Um modelo para geração de carga de servidores Web utilizando classificação de conteúdo**. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES, 2006.

[2] KATSAROS, K. V.; XYLOMENOS, G.; POLYZOS, G. C. **GlobeTraff**: A Traffic Workload Generator for the Performance Evaluation of Future Internet Architectures. In: PROCEEDINGS OF THE IFIP INTERNATIONAL CONFERENCE ON NEW TECHNOLOGIES, MOBILITY AND SECURITY, 2012.

[3] BARFORD, P; CROVELLA, M. **Generating Representative Web Workloads for Network and Server Performance Evaluation**. In: PROCEEDINGS OF THE 1998 ACM SIGMETRICS INTERNATIONAL CONFERENCE ON MEASUREMENT AND MODELING OF COMPUTER SYSTEMS, 1998.

[4] PRIES, R; MAGYARI, Z; TRAN-GIA, P. **An HTTP Web Traffic Model Based on the Top One Million Visited Web Pages**. In: 8TH EURO-NF CONFERENCE ON NEXT GENERATION INTERNET, 2012.

[5] WIKIPEDIA. **Hypertext Transfer Protocol**. Disponível em: <http://pt.wikipedia.org/wiki/HTTP>. Acesso em: 03/07/2013.

[6] INDIANA UNIVERSITY. **What is a proxy server**. Disponível em: <http://kb.iu.edu/data/ahoo.html>. Acesso em: 03/07/2013.

[7] CHOI, H. K; LIMB, J. O. **A Behavioral Model of Web Traffic**. In: INTERNATIONAL CONFERENCE OF NETWORKING PROTOCOL 99 (ICNP 99), TORONTO, CANADA (1999) 327–334.

[8] WIKIPEDIA. **Era Unix**. Disponível em: <http://pt.wikipedia.org/wiki/Era_Unix>. Acesso em: 04/07/2013.

[9] USA TODAY. **Understanding and using Zulu time**. Disponível em: <http://usatoday30.usatoday.com/weather/zulu.htm>. Acesso em: 04/07/2013

[10] INFO WESTER. **Endereço IP**. Disponível em: <http://www.infowester.com/ip.php>. Acesso em: 04/07/2013

[11] IETF. **RFC 1738**: Uniform Resource Locators (URL). Disponível em: <http://www.ietf.org/rfc/rfc1738.txt>. Acesso em: 04/07/2013

[12] SQUID-CACHE WIKI. **Squid native access.log format in detail**. <http://wiki.squid-cache.org/Features/LogFormat>. Acesso em: 06/07/2013

[13] GOLDBERG, A; BUFF, R; SCHMITT, A. **A Comparison of HTTP and HTTPS Performance**. Computer Science Department, Courant Institute of Mathematical Science, New York University

[14] INDIANA UNIVERSITY. **What is FTP**. Disponível em: <http://kb.iu.edu/data/aerg.html>. Acesso em: 06/07/2013

[15] BATISTA, I. L; SALVI, R. F; LUCAS, L. B. **Modelos Científicos e Suas Relações Com a Epistemologia Da Ciência e a Educação Científica**. IN: VIII ENPEC, 2011.

[16] WIKIPEDIA. **Modelo científico**. Disponível em <http://pt.wikipedia.org/wiki/Modelo_cientifico>. Acesso em: 07/07/2013

[17] ABOUT. **What is Perl**. Disponível em: <http://perl.about.com/od/gettingstartedwithperl/p/whatisperl.htm>. Acesso em: 11/07/2013.

[18] CHRISTIAS, P. **grep**. Man-cgi 1.15, 1994. Disponível em: < http://unixhelp.ed.ac.uk/CGI/man-cgi?grep>. Acesso em: 11/07/2013.

[19] R-PROJECT. **What is R**. Disponível em: <http://www.r-project.org/>. Acesso em: 16/07/2013

[20] CUNHA, C; BESTAVROS, A; CROVELLA, M. **Characteristics of WWW Client-based Traces**. Boston University Technical Report, BUCS-95-010, 1995.

[21] Barford, P; Bestavros, A; Bradley, A; Crovella, M. **Changes in Web Client Access Patterns**: Characteristics and Caching, Special Issue on Characterization and Performance Evaluation, Vol. 2, p. 15-28, 1999

[22] LUCAMBIO, F. **Gráfico Q-Q plot**. Disponível em: <http://people.ufpr.br/~lucambio/MReg/qqplot.pdf>. Acesso em: 18/07/2013

# APÊNDICES

## APÊNDICE A – Script para identificar os servidores com maior número de transferências realizadas

```
#!/usr/bin/perl
use strict;
use POSIX qw(strftime);
use IO::Handle;
use List::MoreUtils qw(firstidx);
use DBD::Pg;

open ARQUIVO,"<$ARGV[0]";
my @trace = <ARQUIVO>;

my $my_host = "localhost";
my $my_database = "db";
my $my_user = "script";
my $my_password = "script";
my $myh = DBI->connect("DBI:mysql:dbname=$my_database;host=$my_host", "$my_user",
"$my_password" , {'RaiseError' => 1});

my (@site,@quantidade_de_acessos);
my $controle = 0;
my $total = $#trace;
foreach my $linha (@trace){
    $controle++;
    if ($linha =~ /GET http:\/\/(www\.)?(.[^\/]*)/){
        my $endereco = $2;
        $myh->do("insert into temp_proxy_2(site) values ('$endereco')");
        print "1 - $controle/$total\n";
    }
}
$myh->disconnect();
```

## APÊNDICE B – Script para separar os *logs* de cada servidor em sessões

```
#!/usr/bin/perl
$| = 1;
my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    open ARQUIVO,"<$diretorio_relevante/logs-$diretorio_relevante";
    my $primeiro_tempo;
    foreach my $linha (<ARQUIVO>){
        my @divisao = split(" ",$linha);
        my $TEMPO = @divisao[0];
        my $TAMANHO = @divisao[4];
        my $IP = @divisao[2];
        my $URL = @divisao[6];
        my $MIME_TYPE = @divisao[$#divisao];
        my $conteudo;
        if ($URL =~ m/.+\/\w+\.(\w+)$/){
            $conteudo = $1;
        }
        elsif ($URL =~ m/.+\/.+\.(gif|jpg|jpeg|js|css|png|swf|htm|html)/){
            $conteudo = $1;
        }
        elsif ($MIME_TYPE =~ m/.+\/(.+)$/){
            $conteudo = $1;
        }
        else{
            $conteudo = "html";
        }
        $conteudo =~ tr/[A-Z]/[a-z]/;
```

```
        $conteudo = "html" if ($conteudo eq "com");
        $conteudo = "html" if ($conteudo eq "htm");
        $conteudo = "jpg" if ($conteudo eq "jpeg");
        $conteudo = "javascript" if ($conteudo eq "js" or $conteudo eq "x-javascript");
        $conteudo = "swf" if ($conteudo eq "x-shockwave-flash");
        $conteudo = "outros" if ($conteudo ne "html" and $conteudo ne "gif" and $conteudo ne
"javascript" and $conteudo ne "jpg" and $conteudo ne "css" and $conteudo ne "png" and
$conteudo ne "swf");

        open SESSAO,">>$diretorio_relevante/sessoes/sessao_$IP";
        print SESSAO "$TEMPO $TAMANHO $conteudo\n";
        close SESSAO;
    }
}
```

## APÊNDICE C – Script para verificar quais as classes de arquivos que foram mais transferidas

```
#!/usr/bin/perl
use DBD::Pg;

my $my_host = "localhost";
my $my_database = "db";
my $my_user = "script";
my $my_password = "script";
my $myh = DBI->connect("DBI:mysql:dbname=$my_database;host=$my_host", "$my_user",
"$my_password" , {'RaiseError' => 1});

open ARQUIVO,"<$ARGV[0]";
foreach my $linha (<ARQUIVO>){
    my @divisao = split(" ",$linha);
    my $URL = @divisao[6];
    my $MIME_TYPE = @divisao[$#divisao];
    my $conteudo;
    if ($URL =~ m/.+\/\w+\.(\w+)$/){
        $conteudo = $1;
    }
    elsif ($MIME_TYPE =~ m/.+\/(.+)$/){
        $conteudo = $1;
    }
    else{
        $conteudo = "html";
    }
    $conteudo =~ tr/[A-Z]/[a-z]/;
    $conteudo = "html" if ($conteudo eq "com");
    $myh->do("insert into temp_proxy(string) values ('$conteudo')");
}
$myh->disconnect();
```

APÊNDICE D – Script para separar os *logs* de cada servidor em arquivos contendo sessões de único usuário, agrupando os tipos de arquivos em classes

```
#!/usr/bin/perl
$| = 1;
my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    open ARQUIVO,"<$diretorio_relevante/logs-$diretorio_relevante";
    my $primeiro_tempo;
    foreach my $linha (<ARQUIVO>){
        my @divisao = split(" ",$linha);
        my $TEMPO = @divisao[0];
        my $TAMANHO = @divisao[4];
        my $IP = @divisao[2];
        my $URL = @divisao[6];
        my $MIME_TYPE = @divisao[$#divisao];
        my $conteudo;
        if ($URL =~ m/.+\/\w+\.(\w+)$/){
            $conteudo = $1;
        }
        elsif ($URL =~ m/.+\/.+\.(gif|jpg|jpeg|js|css|png|swf|htm|html)/){
            $conteudo = $1;
        }
        elsif ($MIME_TYPE =~ m/.+\/(.+)$/){
            $conteudo = $1;
        }
        else{
            $conteudo = "html";
        }
        $conteudo =~ tr/[A-Z]/[a-z]/;
        $conteudo = "html" if ($conteudo eq "com");
        $conteudo = "html" if ($conteudo eq "htm");
        $conteudo = "jpg" if ($conteudo eq "jpeg");
        $conteudo = "javascript" if ($conteudo eq "js" or $conteudo eq "x-javascript");
        $conteudo = "swf" if ($conteudo eq "x-shockwave-flash");
        $conteudo = "outros" if ($conteudo ne "html" and $conteudo ne "gif" and $conteudo ne
"javascript" and $conteudo ne "jpg" and $conteudo ne "css" and $conteudo ne "png" and
$conteudo ne "swf");

        open SESSAO,">>$diretorio_relevante/sessoes/sessao_$IP";
        print SESSAO "$TEMPO $TAMANHO $conteudo\n";
        close SESSAO;
    }
}
```

APÊNDICE E – Script para parametrizar a matriz de transição (modelo A) e os intervalos de OFF, *active-off* e quantidade de requisições por sessão

```
#!/usr/bin/perl
$| = 1;


our $intervalo_entre_requisicoes = 20;

my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    our ($inicio_html, $inicio_gif, $inicio_javascript, $inicio_jpg, $inicio_css, $inicio_png,
$inicio_swf, $inicio_outros, $inicio_fim) = (0,0,0,0,0,0,0,0,0);
    our ($html_html, $html_gif, $html_javascript, $html_jpg, $html_css, $html_png, $html_swf,
$html_outros, $html_fim) = (0,0,0,0,0,0,0,0,0);
    our ($gif_html, $gif_gif, $gif_javascript, $gif_jpg, $gif_css, $gif_png, $gif_swf,
$gif_outros, $gif_fim) = (0,0,0,0,0,0,0,0,0);
    our ($javascript_html, $javascript_gif, $javascript_javascript, $javascript_jpg,
$javascript_css, $javascript_png, $javascript_swf, $javascript_outros, $javascript_fim) =
(0,0,0,0,0,0,0,0,0);
    our ($jpg_html, $jpg_gif, $jpg_javascript, $jpg_jpg, $jpg_css, $jpg_png, $jpg_swf,
$jpg_outros, $jpg_fim) = (0,0,0,0,0,0,0,0,0);
    our ($css_html, $css_gif, $css_javascript, $css_jpg, $css_css, $css_png, $css_swf,
$css_outros, $css_fim) = (0,0,0,0,0,0,0,0,0);
    our ($png_html, $png_gif, $png_javascript, $png_jpg, $png_css, $png_png, $png_swf,
$png_outros, $png_fim) = (0,0,0,0,0,0,0,0,0);
    our ($swf_html, $swf_gif, $swf_javascript, $swf_jpg, $swf_css, $swf_png, $swf_swf,
$swf_outros, $swf_fim) = (0,0,0,0,0,0,0,0,0);
    our ($outros_html, $outros_gif, $outros_javascript, $outros_jpg, $outros_css, $outros_png,
$outros_swf, $outros_outros, $outros_fim) = (0,0,0,0,0,0,0,0,0);

    our
($total_inicio,$total_html,$total_gif,$total_javascript,$total_jpg,$total_css,$total_png,$tota
l_swf,$total_outros) = (0,0,0,0,0,0,0,0,0);
    our $flag_inicio = 0;
    open INTERVALO_OFF,">$diretorio_relevante/sessoes/intervalo_off.csv";
    open INTERVALO_ACTIVE_OFF,">$diretorio_relevante/sessoes/intervalo_active_off.csv";
    my @arquivos = `dir "$diretorio_relevante/sessoes"`;
    foreach my $arquivo(@arquivos){
        if ($arquivo =~ /.+\s.+\s.+\s(sessao_\S+)/){
            my $arquivo_relevante = $1;
            open ARQUIVO,"<$diretorio_relevante/sessoes/$arquivo_relevante";
            my @linhas = <ARQUIVO>;
            for (my $x = 1; $x <= $#linhas; $x++){
                my @divisao_linha_anterior = split(" ",@linhas[$x-1]);
                my @divisao_linha_atual = split(" ",@linhas[$x]);

                $flag_inicio = 1 if (!$flag_inicio && (@divisao_linha_atual[0]-
@divisao_linha_anterior[0]) >= $intervalo_entre_requisicoes);

                if ($flag_inicio){
                    if ((@divisao_linha_atual[0]-@divisao_linha_anterior[0]) > 0){
                        if ((@divisao_linha_atual[0]-@divisao_linha_anterior[0]) >=
$intervalo_entre_requisicoes){
                            $tempo = sprintf("%.5f",@divisao_linha_atual[0]-
@divisao_linha_anterior[0]);
                            print INTERVALO_OFF $tempo . "\n";
                            $var = @divisao_linha_anterior[2] . "_fim";
                            $$var++;
                            my $var_total = "total_" . @divisao_linha_anterior[2];
                            $$var_total++;
                            $var = "inicio_" . @divisao_linha_atual[2];
                            $$var++;
                            my $var_total = "total_inicio";
                            $$var_total++;
```

```
                        }
                        else{
                            $tempo = sprintf("%.5f",@divisao_linha_atual[0]-
@divisao_linha_anterior[0]);
                            print INTERVALO_ACTIVE_OFF $tempo . "\n";
                            my $var = @divisao_linha_anterior[2] . "_" .
@divisao_linha_atual[2];
                            $$var++;
                            my $var_total = "total_" . @divisao_linha_anterior[2];
                            $$var_total++;
                        }
                    }
                }
            }
        }
    }
    close INTERVALO_OFF;
    close INTERVALO_ACTIVE_OFF;
    foreach ($inicio_html, $inicio_gif, $inicio_javascript, $inicio_jpg, $inicio_css,
$inicio_png, $inicio_swf, $inicio_outros, $inicio_fim) {$_ = sprintf("%.4f",$_/$total_inicio)
if $total_inicio;}
    foreach ($html_html, $html_gif, $html_javascript, $html_jpg, $html_css, $html_png,
$html_swf, $html_outros, $html_fim) {$_ = sprintf("%.4f",$_ / $total_html) if $total_html;}
    foreach ($gif_html, $gif_gif, $gif_javascript, $gif_jpg, $gif_css, $gif_png, $gif_swf,
$gif_outros, $gif_fim) {$_ = sprintf("%.4f",$_ / $total_gif) if $total_gif;}
    foreach ($javascript_html, $javascript_gif, $javascript_javascript, $javascript_jpg,
$javascript_css, $javascript_png, $javascript_swf, $javascript_outros, $javascript_fim) {$_ =
sprintf("%.4f",$_ / $total_javascript) if $total_javascript;}
    foreach ($jpg_html, $jpg_gif, $jpg_javascript, $jpg_jpg, $jpg_css, $jpg_png, $jpg_swf,
$jpg_outros, $jpg_fim) {$_ = sprintf("%.4f",$_ / $total_jpg) if $total_jpg;}
    foreach ($css_html, $css_gif, $css_javascript, $css_jpg, $css_css, $css_png, $css_swf,
$css_outros, $css_fim) {$_ = sprintf("%.4f",$_ / $total_css) if $total_css;}
    foreach ($png_html, $png_gif, $png_javascript, $png_jpg, $png_css, $png_png, $png_swf,
$png_outros, $png_fim) {$_ = sprintf("%.4f",$_ / $total_png) if $total_png;}
    foreach ($swf_html, $swf_gif, $swf_javascript, $swf_jpg, $swf_css, $swf_png, $swf_swf,
$swf_outros, $swf_fim) {$_ = sprintf("%.4f",$_ / $total_swf) if $total_swf;}
    foreach ($outros_html, $outros_gif, $outros_javascript, $outros_jpg, $outros_css,
$outros_png, $outros_swf, $outros_outros, $outros_fim) {$_ = sprintf("%.4f",$_ /
$total_outros) if $total_outros;}
    open
MATRIZ,">$diretorio_relevante/sessoes/matriz_transicao_intervalo_$intervalo_entre_requisicoes.
csv";
    print MATRIZ
"$inicio html;$inicio gif;$inicio javascript;$inicio jpg;$inicio css;$inicio png;$inicio swf;$
inicio_outros;$inicio_fim\n";
    print MATRIZ
"$html html;$html gif;$html javascript;$html jpg;$html css;$html png;$html swf;$html outros;$h
tml_fim\n";
    print MATRIZ
"$gif html;$gif gif;$gif javascript;$gif jpg;$gif css;$gif png;$gif swf;$gif outros;$gif fim\n
";
    print MATRIZ
"$javascript html;$javascript gif;$javascript javascript;$javascript jpg;$javascript css;$java
script_png;$javascript_swf;$javascript_outros;$javascript_fim\n";
    print MATRIZ
"$jpg html;$jpg gif;$jpg javascript;$jpg jpg;$jpg css;$jpg png;$jpg swf;$jpg outros;$jpg fim\n
";
    print MATRIZ
"$css html;$css gif;$css javascript;$css jpg;$css css;$css png;$css swf;$css outros;$css fim\n
";
    print MATRIZ
"$png_html;$png_gif;$png_javascript;$png_jpg;$png_css;$png_png;$png_swf;$png_outros;$png_fim\n
";
    print MATRIZ
"$swf html;$swf gif;$swf javascript;$swf jpg;$swf css;$swf png;$swf swf;$swf outros;$swf fim\n
";
    print MATRIZ
"$outros_html;$outros_gif;$outros_javascript;$outros_jpg;$outros_css;$outros_png;$outros_swf;$
outros_outros;$outros_fim\n";
    close MATRIZ;
}
```

APÊNDICE F – Script para separar os tamanhos de cada classe, de cada servidor, em um arquivo por classe

```
#!/usr/bin/perl
$| = 1;
my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    open ARQUIVO,"<$diretorio_relevante/logs-$diretorio_relevante";
    my $primeiro_tempo;
    foreach my $linha (<ARQUIVO>){
        my @divisao = split(" ",$linha);
        my $TEMPO = @divisao[0];
        my $TAMANHO = @divisao[4];
        my $IP = @divisao[2];
        my $URL = @divisao[6];
        my $MIME_TYPE = @divisao[$#divisao];
        my $conteudo;
        if ($URL =~ m/.+\/\w+\.(\w+)$/){
            $conteudo = $1;
        }
        elsif ($URL =~ m/.+\/.+\.(gif|jpg|jpeg|js|css|png|swf|htm|html)/){
            $conteudo = $1;
        }
        elsif ($MIME_TYPE =~ m/.+\/(.+)$/){
            $conteudo = $1;
        }
        else{
            $conteudo = "html";
        }
        $conteudo =~ tr/[A-Z]/[a-z]/;
        $conteudo = "html" if ($conteudo eq "com");
        $conteudo = "html" if ($conteudo eq "htm");
        $conteudo = "jpg" if ($conteudo eq "jpeg");
        $conteudo = "javascript" if ($conteudo eq "js" or $conteudo eq "x-javascript");
        $conteudo = "swf" if ($conteudo eq "x-shockwave-flash");
        $conteudo = "outros" if ($conteudo ne "html" and $conteudo ne "gif" and $conteudo ne
"javascript" and $conteudo ne "jpg" and $conteudo ne "css" and $conteudo ne "png" and
$conteudo ne "swf");

        open TAMANHO,">>$diretorio_relevante/tamanho_conteudo/tamanho_$conteudo";
        print TAMANHO "$TAMANHO\n";
        close TAMANHO;
    }
}
```

APÊNDICE G – Script para gerar arquivos contendo tamanhos dos objetos principais, tamanhos dos objetos embutidos e número de objetos embutidos por requisição

```
#!/usr/bin/perl
$| = 1;

our $intervalo_entre_requisicoes = 20;

my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    our $flag_inicio = 0;
    our $flag_inicio_2 = 0;
    open MAIN_OBJECT_SIZE,">$diretorio_relevante/sessoes/modeloB_main_object_size.csv";
    open INLINE_OBJECT_SIZE,">$diretorio_relevante/sessoes/modeloB_inline_object_size.csv";
    open
INLINE_OBJECT_NUMBER,">$diretorio_relevante/sessoes/modeloB_inline_object_number.csv";
    our $numero_de_inline_objects = 0;
    my @arquivos = `dir "$diretorio_relevante/sessoes"`;
    foreach my $arquivo(@arquivos){
        if ($arquivo =~ /.+\s.+\s.+\s(sessao_\S+)/){
            my $arquivo_relevante = $1;
            open ARQUIVO,"<$diretorio_relevante/sessoes/$arquivo_relevante";
            my @linhas = <ARQUIVO>;
            for (my $x = 1; $x <= $#linhas; $x++){
                my @divisao_linha_anterior = split(" ",@linhas[$x-1]);
                my @divisao_linha_atual = split(" ",@linhas[$x]);

                $flag_inicio = 1 if (!$flag_inicio && (@divisao_linha_atual[0]-
@divisao_linha_anterior[0]) >= $intervalo_entre_requisicoes);

                if ($flag_inicio){
                    if ((@divisao_linha_atual[0]-@divisao_linha_anterior[0]) > 0){
                        if ((@divisao_linha_atual[0]-@divisao_linha_anterior[0]) >=
$intervalo_entre_requisicoes){
                            print MAIN_OBJECT_SIZE "@divisao_linha_atual[1]\n";
                            print INLINE_OBJECT_NUMBER "$numero_de_inline_objects\n" if
($flag_inicio_2);
                            $numero_de_inline_objects = 0;
                        }
                        else{
                            print INLINE_OBJECT_SIZE "@divisao_linha_atual[1]\n";
                            $numero_de_inline_objects++;
                            $flag_inicio_2 = 1 if (!$flag_inicio_2);
                        }
                    }
                }
            }
        }
    }
    close MAIN_OBJECT_SIZE;
    close INLINE_OBJECT_SIZE;
    close INLINE_OBJECT_NUMBER;
}
```

# APÊNDICE H – Script para simular sessões com o modelo A

```
#!/usr/bin/perl
$| = 1;

my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    my $site = $diretorio_relevante;

    # CONTAR QUANTAS SESSÕES O SITE TEM (TRACE REAL)
    my $count_numero_sessoes_reais = 0;
    my @sessoes_reais = `dir "$diretorio_relevante\\sessoes"`;
    foreach my $arquivo(@sessoes_reais){
        if ($arquivo =~ /sessao_10/){
            $count_numero_sessoes_reais++;
        }
    }

    my @todas_classes =
("inicio","html","gif","javascript","jpg","css","png","swf","outros","fim");

    # CARREGAR MATRIZ DE TRANSIÇÃO
    my @matriz_de_transicoes;
    open MATRIZ,"<$site/sessoes/matriz_transicao_intervalo_20.csv";
    my @arquivo_matriz = <MATRIZ>;
    for (my $x = 0; $x <= $#arquivo_matriz; $x++){
        my @coluna = split(";",@arquivo_matriz[$x]);
        for (my $y = 0; $y <= $#coluna; $y++){
            my ($probabilidade_de,$probabilidade_para);
            if ($y == 0){
                $probabilidade_de = 0;
                $probabilidade_para = @coluna[$y];
            }
            else{
                $probabilidade_de = @matriz_de_transicoes[$#matriz_de_transicoes]-
>{probabilidade_para};
                $probabilidade_para = @matriz_de_transicoes[$#matriz_de_transicoes]-
>{probabilidade_para} + @coluna[$y];
            }
            push(@matriz_de_transicoes,nova
transicao(@todas_classes[$x],@todas_classes[$y+1],$probabilidade_de,$probabilidade_para));
        }
    }

    # CARREGAR PARAMETROS DAS DISTRIBUIÇÕES DE TAMANHO DOS ARQUIVOS (Distribuição de Pareto
Tipo II)
    my @classes;
    open PARAMETROS,"<$site/tamanho_conteudo/parametros_script.csv";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/,/./; $linha =~ s/,/./;
        my ($nome,$alfa,$beta) = split(";",$linha);
        push (@classes, nova classe($nome,$alfa,$beta));
    }

    # CARREGAR PARAMETROS DO INTERVALO OFF (Distribuição de Weibull)
    my ($intervalo_off_shape, $intervalo_off_scale);
    open PARAMETROS,"<$site/sessoes/parametros_intervalo_off";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $intervalo_off_shape = $valor if $atributo eq "shape";
        $intervalo_off_scale = $valor if $atributo eq "scale";
    }
```

```
    # CARREGAR PARAMETROS DO INTERVALO ACTIVE-OFF (Distribuição de Weibull)
    my ($intervalo_active_off_shape, $intervalo_active_off_scale);
    open PARAMETROS,"<$site/sessoes/parâmetros_intervalo_active_off";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $intervalo_active_off_shape = $valor if $atributo eq "shape";
        $intervalo_active_off_scale = $valor if $atributo eq "scale";
    }

    # CARREGAR PARAMETROS DA QUANTIDADE DE REQUISIÇÕES POR SESSÃO (Distribuição Exponencial)
    my ($quantidade_de_requisicoes_rate);
    open PARAMETROS,"<$site/sessoes/parametros_quantidade_requisicoes_por_sessao";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $quantidade_de_requisicoes_rate = $valor if $atributo eq "rate";
    }

    for my $num (1..$count_numero_sessoes_reais){

        open OUTPUT,">$site/sessoes_simuladas_modeloA/sessao_simulada_$num";

        # NÚMERO DE REQUISICOES
        my $rate = $quantidade_de_requisicoes_rate;
         # EXPONENCIAL
        my $numero_de_requisicoes = sprintf("%.0f",(-log(1-rand())/log(exp(1)))/$rate);

        # INTERVALO OFF
        my $scale = $intervalo_off_scale;
        my $shape = $intervalo_off_shape;
        my $intervalo_inicio = $scale*(-log(1-rand())/log(exp(1)))**(1/$shape); # WEIBULL

        our $tempo = $intervalo_inicio;

        our $classe_atual = "inicio";

        for (my $requisicao_atual = 0; $requisicao_atual <= $numero_de_requisicoes;
$requisicao_atual++){
            requisicao: while(1){
                if ($classe_atual eq "fim"){
                    # INTERVALO OFF
                    my $intervalo = $scale*(-log(1-rand())/log(exp(1)))**(1/$shape); # WEIBULL
                    $tempo += $intervalo;

                    $classe_atual = "inicio";
                }
                else{
                    # INTERVALO ACTIVE OFF
                    my $scale_aoff = $intervalo_active_off_scale;
                    my $shape_aoff = $intervalo_active_off_shape;
                    # WEIBULL
                    my $intervalo = $scale_aoff*(-log(1-rand())/log(exp(1)))**(1/$shape_aoff);
                    $tempo += $intervalo;
                }
                my $aleatorio = rand();
                foreach my $transicao (@matriz_de_transicoes){
                    if ($transicao->{classe_de} eq $classe_atual && $aleatorio > $transicao-
>{probabilidade_de} && $aleatorio <= $transicao->{probabilidade_para}){
                    # MATRIZ DE TRANSIÇÃO
                        $classe_atual = $transicao->{classe_para};

                        last requisicao if ($classe_atual eq "fim");

                        # TAMANHO DO ARQUIVO
                        my ($alfa,$beta);
                        foreach my $classe(@classes){
                            if ($classe->{nome} eq $classe_atual){
                                $alfa = $classe->{alfa};
                                $beta = $classe->{beta};
                                last;
                            }
                        }
```

```
                        # PARETO TIPO 2
                        my $tamanho = sprintf("%.0f", -$beta*(1 - rand()**(-1/$alfa)));
                        my $out = "$tempo $tamanho\n";
                        $out =~ s/\./\,/g; # substituindo pontos por vírgulas
                        print OUTPUT $out;

                        last;
                    }
                }
            }
        }
        close OUTPUT;
        print "\n\n";

    }
}
sub trim {
  my $string = shift;
  for ($string) {
    s/^\s+//;
    s/\s+$//;
  }
  return $string;
}

sub get_time {
    my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime(time);
    if ($min < 10){
        $min = "0" . $min;
    }
    $year=$year+1900;
    $mon++;
    return $mday . "-" . $mon . "-" . $year . "_" . $hour . "-" . $min . "-" . $sec;
}

package transicao;
sub nova
{
    my $classe = shift;
    my $self = {
        classe_de => shift,
        classe_para => shift,
        probabilidade_de => shift,
        probabilidade_para => shift,
    };
    bless $self, $classe;
    return $self;
}

package classe;
sub nova
{
    my $classe = shift;
    my $self = {
        nome => shift,
        alfa => shift,
        beta => shift,
    };
    bless $self, $classe;
    return $self;
}
```

# APÊNDICE I – Script para simular sessões com o modelo B

```
#!/usr/bin/perl
$| = 1;

my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    my $site = $diretorio_relevante;

    # CONTAR QUANTAS SESSÕES O SITE TEM (TRACE REAL)
    my $count_numero_sessoes_reais = 0;
    my @sessoes_reais = `dir "$diretorio_relevante\\sessoes"`;
    foreach my $arquivo(@sessoes_reais){
        if ($arquivo =~ /sessao_10/){
            $count_numero_sessoes_reais++;
        }
    }

    # CARREGAR PARAMETROS DO TAMANHO DO "MAIN OBJECT" (Distribuição de Pareto Tipo II)
    my ($main_object_size_alfa, $main_object_size_beta);
    open PARAMETROS,"<$site/sessoes/parametros_modeloB_main_object_size";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $main_object_size_alfa = $valor if $atributo eq "shape";
        $main_object_size_beta = $valor if $atributo eq "scale";
    }

    # CARREGAR PARAMETROS DO TAMANHO DO "INLINE OBJECT" (Distribuição de Pareto Tipo II)
    my ($inline_object_size_alfa, $inline_object_size_beta);
    open PARAMETROS,"<$site/sessoes/parametros_modeloB_inline_object_size";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $inline_object_size_alfa = $valor if $atributo eq "shape";
        $inline_object_size_beta = $valor if $atributo eq "scale";
    }

    # CARREGAR PARAMETROS DA QUANTIDADE DE "INLINE OBJECT" (Distribuição Exponencial)
    my ($inline_object_number_rate);
    open PARAMETROS,"<$site/sessoes/parametros_modeloB_inline_object_number";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $inline_object_number_rate = $valor if $atributo eq "rate";
    }

    # CARREGAR PARAMETROS DO INTERVALO OFF (Distribuição de Weibull)
    my ($intervalo_off_shape, $intervalo_off_scale);
    open PARAMETROS,"<$site/sessoes/parametros_intervalo_off";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $intervalo_off_shape = $valor if $atributo eq "shape";
        $intervalo_off_scale = $valor if $atributo eq "scale";
    }

    # CARREGAR PARAMETROS DO INTERVALO ACTIVE-OFF (Distribuição de Weibull)
    my ($intervalo_active_off_shape, $intervalo_active_off_scale);
    open PARAMETROS,"<$site/sessoes/parametros_intervalo_active_off";
```

```
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $intervalo_active_off_shape = $valor if $atributo eq "shape";
        $intervalo_active_off_scale = $valor if $atributo eq "scale";
    }

    # CARREGAR PARAMETROS DA QUANTIDADE DE REQUISIÇÕES POR SESSÃO (Distribuição Exponencial)
    my ($quantidade_de_requisicoes_rate);
    open PARAMETROS,"<$site/sessoes/parametros_quantidade_requisicoes_por_sessao";
    my @arquivo_parametros = <PARAMETROS>;
    foreach (@arquivo_parametros){
        my $linha = trim($_);
        $linha =~ s/\,/\./g;
        my ($atributo,$valor) = split(" = ",$linha);
        $quantidade_de_requisicoes_rate = $valor if $atributo eq "rate";
    }

    for my $num (1..$count_numero_sessoes_reais){
        open OUTPUT,">$site/sessoes_simuladas_modeloB/sessao_simulada_modeloB_$num";

        # INTERVALO OFF
        my $scale = $intervalo_off_scale;
        my $shape = $intervalo_off_shape;
        my $intervalo_inicio = $scale*(-log(1-rand())/log(exp(1)))**(1/$shape); # WEIBULL

        our $tempo = $intervalo_inicio;

        # NÚMERO DE REQUISICOES
        my $rate = $quantidade_de_requisicoes_rate;
        my $numero_de_requisicoes = sprintf("%.0f",(-log(1-rand())/log(exp(1)))/$rate);     #
EXPONENCIAL

        for (my $requisicao_atual = 0; $requisicao_atual <= $numero_de_requisicoes;
$requisicao_atual++){

            # TAMANHO DO MAIN OBJECT
            my $alfa = $main_object_size_alfa;
            my $beta = $main_object_size_beta;
            my $main_object_size = sprintf("%.0f", -$beta*(1 - rand()**(-1/$alfa)));        #
PARETO TIPO 2

            my $out = "$tempo $main_object_size\n";
            $out =~ s/\./\,/g;
            print OUTPUT $out;

            # NÚMERO DE INLINE OBJECTS
            my $rate = $inline_object_number_rate;
            my $inline_objects_number = sprintf("%.0f",(-log(1-rand())/log(exp(1)))/$rate+1);
# EXPONENCIAL

            for (1..$inline_objects_number){

                # INTERVALO ACTIVE OFF
                my $scale_aoff = $intervalo_active_off_scale;
                my $shape_aoff = $intervalo_active_off_shape;
                my $intervalo = $scale_aoff*(-log(1-rand())/log(exp(1)))**(1/$shape_aoff);  #
WEIBULL
                $tempo += $intervalo;

                # TAMANHO DO INLINE OBJECT
                my $alfa = $inline_object_size_alfa;
                my $beta = $inline_object_size_beta;
                my $inline_object_size = sprintf("%.0f", -$beta*(1 - rand()**(-1/$alfa)));
# PARETO TIPO 2

                my $out = "$tempo $inline_object_size\n";
                $out =~ s/\./\,/g;
                print OUTPUT $out;
            }

            # INTERVALO OFF
            my $intervalo = $scale*(-log(1-rand())/log(exp(1)))**(1/$shape);    # WEIBULL
            $tempo += $intervalo;
        }
```

```
        close OUTPUT;
        print "\n\n";

    }
}

sub trim {
  my $string = shift;
  for ($string) {
    s/^\s+//;
    s/\s+$//;
  }
  return $string;
}
```

APÊNDICE J – Script para somar os tamanhos de arquivos transferidos em intervalos de 1 segundo

```
#!/usr/bin/perl
$| = 1;
use strict;
my @diretorios = `dir`;
my @diretorios_relevantes;
foreach my $diretorio(@diretorios){
    if ($diretorio =~ /<DIR>\s+(\S+\.\S+\.\S+)/){
        push(@diretorios_relevantes,$1);
    }
}
foreach my $diretorio_relevante (@diretorios_relevantes){
    my $site = $diretorio_relevante;
    print "site: $site\n";

    # ARQUIVO COM TRACE REAL DO SITE
    my $trace_real_completo = $diretorio_relevante."/logs-".$diretorio_relevante;

    # ENCONTRAR ARQUIVO COM CONCATENAÇÃO DAS SESSÕES SIMULADAS - MODELO A
    my $todas_sessoes_modeloA;
    my @sessoes = `dir "$diretorio_relevante\\sessoes_simuladas_modeloA"`;
    foreach my $arquivo(@sessoes){
        if ($arquivo =~ /(\S+_sessoes_modeloA)/){
            $todas_sessoes_modeloA = $diretorio_relevante."/sessoes_simuladas_modeloA/".$1;
        }
    }

    # ENCONTRAR ARQUIVO COM CONCATENAÇÃO DAS SESSÕES SIMULADAS - MODELO B
    my $todas_sessoes_modeloB;
    my @sessoes = `dir "$diretorio_relevante\\sessoes_simuladas_modeloB"`;
    foreach my $arquivo(@sessoes){
        if ($arquivo =~ /(\S+_sessoes_modeloB)/){
            $todas_sessoes_modeloB = $diretorio_relevante."/sessoes_simuladas_modeloB/".$1;
        }
    }

    # ENCONTRAR SESSÃO DO TRACE REAL COM MAIOR NÚMERO DE TRANSFERÊNCIAS
    my $maior_sessao_real;
    my @sessoes = `dir "$diretorio_relevante\\sessoes" /o:s`;
    foreach my $arquivo(@sessoes){
        if ($arquivo =~ /(sessao_10\S+)/){
            $maior_sessao_real = $diretorio_relevante."/sessoes/".$1;
        }
    }

    # ENCONTRAR SESSÃO DAS SIMULAÇÕES DO MODELO A COM MAIOR NÚMERO DE TRANSFERÊNCIAS
    my $maior_sessao_modeloA;
    my @sessoes = `dir "$diretorio_relevante\\sessoes_simuladas_modeloA" /o:s`;
    foreach my $arquivo(@sessoes){
        if ($arquivo =~ /(sessao_simulada_\S+)/){
            $maior_sessao_modeloA = $diretorio_relevante."/sessoes_simuladas_modeloA/".$1;
        }
```

```
    }

    # ENCONTRAR SESSÃO DAS SIMULAÇÕES DO MODELO B COM MAIOR NÚMERO DE TRANSFERÊNCIAS
    my $maior_sessao_modeloB;
    my @sessoes = `dir "$diretorio_relevante\\sessoes_simuladas_modeloB" /o:s`;
    foreach my $arquivo(@sessoes){
        if ($arquivo =~ /(sessao_simulada_modeloB_\S+)/){
            $maior_sessao_modeloB = $diretorio_relevante."/sessoes_simuladas_modeloB/".$1;
        }
    }

    foreach my $intervalo (1){
        print "\tintervalo: $intervalo\n";
        open COMANDOS_R,">COMPARACOES/comandos_R-$site-$intervalo\s.r";
        my $flag = 0;
        foreach my $arquivo
($trace_real_completo,$todas_sessoes_modeloA,$todas_sessoes_modeloB,$maior_sessao_real,
$maior_sessao_modeloA, $maior_sessao_modeloB){
            print "\t\tarquivo: ($arquivo)\n";
            $flag++;
            my @split = split("\/",$arquivo);
            my $coluna_tamanho_arquivo;
            my $nome = @split[$#split];
            if ($nome =~ /logs-/){
                $coluna_tamanho_arquivo = 4;
            }
            else{
                $coluna_tamanho_arquivo = 1;
            }

            # CRIANDO LINHAS DE COMANDO PARA O R
            if ($flag == 1){
                print COMANDOS_R 'rm(list=ls(all=TRUE)); jpeg("'.$site.'-multiplas-sessoes-
'.$intervalo.'s.jpeg", width = 900, height = 700, quality=100, type="cairo"); par(cex = 1.5);
s2<-scan("'."serie-$site-$nome-$intervalo\s".'"); acfs2<-acf(s2, plot=FALSE);
plot(acfs2$acf,type="l",col=3,ylim=c(0,1),xlim=c(0,48),main="ACF '.$intervalo.'s - multiplas
sessoes ('.$site.')");';
            }
            elsif ($flag == 2 || $flag == 5){
                print COMANDOS_R 'a2<-scan("'."serie-$site-$nome-$intervalo\s".'"); acfa2<-
acf(a2, plot=FALSE); lines(acfa2$acf,type="l",col=4);';
            }
            elsif ($flag == 3 || $flag == 6){
                print COMANDOS_R 'b2<-scan("'."serie-$site-$nome-$intervalo\s".'"); acfb2<-
acf(b2, plot=FALSE); lines(acfb2$acf,type="l",col=2); dev.off();';
            }
            elsif ($flag == 4){
                print COMANDOS_R 'rm(list=ls(all=TRUE)); jpeg("'.$site.'-sessao-unica-
'.$intervalo.'s.jpeg", width = 900, height = 700, quality=100, type="cairo"); par(cex = 1.5);
s2<-scan("'."serie-$site-$nome-$intervalo\s".'"); acfs2<-acf(s2, plot=FALSE);
plot(acfs2$acf,type="l",col=3,ylim=c(0,1),xlim=c(0,48),main="ACF '.$intervalo.'s - unica
sessao ('.$site.')");';
            }

            open INPUT_ACF,">COMPARACOES/serie-$site-$nome-$intervalo\s";
            open SESSAO,"<$arquivo";
            my @sessao = <SESSAO>;
            foreach (@sessao){
                $_ =~ s/\,/\./g;
            }
            my @primeira_linha = split(" ",@sessao[0]);
            my @ultima_linha = split(" ",@sessao[$#sessao]);

            my $indice_sessao = 0;

            for (my $tempo = @primeira_linha[0]; $tempo <= @ultima_linha[0]; $tempo = $tempo +
$intervalo){
                my $soma_tamanhos = 0;
                for my $x ($indice_sessao..$#sessao){
                    my @linha_atual = split(" ",@sessao[$x]);
                    last if @linha_atual[0] >= $tempo + $intervalo;
                    $soma_tamanhos += @linha_atual[$coluna_tamanho_arquivo] if
(@linha_atual[0] >= $tempo && @linha_atual[0] < $tempo + $intervalo);
                    $indice_sessao++;
                }
                print INPUT_ACF "$soma_tamanhos\n";
            }
```

```
        }
        close COMANDOS_R;
    }
}
```

# APÊNDICE K – Matrizes de probabilidade de transição

SERVIDOR “L.YIMG.COM”

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.0038 | 0.2632 | 0.1851 | 0.2491 | 0.0835 | 0.1051 | 0.1064 | 0.0038 | 0 |
| HTML | 0.3725 | 0.0196 | 0 | 0.0131 | 0 | 0 | 0 | 0.4510 | 0.1438 |
| GIF | 0.0022 | 0.5943 | 0.0352 | 0.1475 | 0.0078 | 0.0698 | 0.0253 | 0.0069 | 0.1110 |
| JPG | 0.0074 | 0.2016 | 0.3065 | 0.1278 | 0.0876 | 0.0678 | 0.0445 | 0.0039 | 0.1529 |
| JAVASCRIPT | 0.0036 | 0.1093 | 0.0471 | 0.6427 | 0.0013 | 0.0834 | 0.0371 | 0.0025 | 0.0731 |
| CSS | 0.0008 | 0.2232 | 0.1832 | 0.0592 | 0.4104 | 0.0608 | 0.0056 | 0.0048 | 0.0520 |
| PNG | 0.0015 | 0.1507 | 0.0587 | 0.1950 | 0.0057 | 0.4841 | 0.0388 | 0.0017 | 0.0637 |
| SWF | 0.0036 | 0.1407 | 0.0427 | 0.2125 | 0.0024 | 0.0291 | 0.3276 | 0.0071 | 0.2344 |
| OUTROS | 0 | 0.2913 | 0.0340 | 0.1262 | 0.0485 | 0.0340 | 0.0243 | 0.1505 | 0.2913 |

SERVIDOR “MAIL.GOOGLE.COM”

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.5621 | 0.1218 | 0.0750 | 0.0105 | 0.0015 | 0.0178 | 0.0019 | 0.2094 | 0 |
| HTML | 0.5896 | 0.0333 | 0.1542 | 0.0194 | 0.0007 | 0.0492 | 0.0186 | 0.0344 | 0.1006 |
| GIF | 0.1470 | 0.4605 | 0.0802 | 0.0129 | 0.0004 | 0.0646 | 0.0065 | 0.0723 | 0.1557 |
| JPG | 0.3984 | 0.0717 | 0.1539 | 0.0149 | 0.0020 | 0.0463 | 0.0509 | 0.0386 | 0.2232 |
| JAVASCRIPT | 0.1758 | 0.0389 | 0.1062 | 0.4855 | 0 | 0.0478 | 0.0053 | 0.0136 | 0.1268 |
| CSS | 0.1867 | 0.0400 | 0.2000 | 0 | 0.4000 | 0 | 0 | 0.0133 | 0.1600 |
| PNG | 0.1267 | 0.0487 | 0.1277 | 0.0215 | 0.0002 | 0.5006 | 0.0237 | 0.1224 | 0.0285 |
| SWF | 0.1555 | 0.0423 | 0.1915 | 0.0013 | 0 | 0.0258 | 0.5047 | 0.0463 | 0.0325 |
| OUTROS | 0.1527 | 0.0314 | 0.0569 | 0.0053 | 0 | 0.0969 | 0.0345 | 0.1737 | 0.4484 |

SERVIDOR “SDP.TERRA.COM.BR”

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.0011 | 0.1501 | 0 | 0.8433 | 0 | 0.0055 | 0 | 0 | 0 |
| HTML | 0 | 0 | 0 | 0.4000 | 0.2000 | 0 | 0 | 0 | 0.4000 |
| GIF | 0.0001 | 0.7613 | 0 | 0.1863 | 0.0001 | 0.0004 | 0 | 0 | 0.0518 |
| JPG | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JAVASCRIPT | 0.0001 | 0.1960 | 0 | 0.7184 | 0.0004 | 0.0035 | 0 | 0.0001 | 0.0816 |
| CSS | 0 | 0.5714 | 0 | 0 | 0 | 0.2857 | 0 | 0 | 0.1429 |
| PNG | 0.0071 | 0 | 0 | 0.3333 | 0 | 0.5957 | 0 | 0 | 0.0638 |
| SWF | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUTROS | 0 | 0 | 0 | 10000 | 0 | 0 | 0 | 0 | 0 |

SERVIDOR “STF.TERRA.COM.BR”

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.1353 | 0.0814 | 0.3650 | 0.0290 | 0.3729 | 0.0005 | 0.0125 | 0.0035 | 0 |
| HTML | 0.1371 | 0.0311 | 0.3035 | 0 | 0.1444 | 0 | 0 | 0.0018 | 0.3821 |
| GIF | 0.0039 | 0.6439 | 0.1258 | 0.1682 | 0.0045 | 0.0143 | 0.0002 | 0.0003 | 0.0388 |
| JPG | 0.0154 | 0.1215 | 0.6761 | 0.0235 | 0.0503 | 0.0017 | 0.0014 | 0.0005 | 0.1096 |
| JAVASCRIPT | 0 | 0.2993 | 0.1042 | 0.5664 | 0.0012 | 0.0134 | 0.0002 | 0 | 0.0153 |
| CSS | 0.0003 | 0.0361 | 0.2376 | 0.0062 | 0.7008 | 0 | 0 | 0 | 0.0190 |
| PNG | 0 | 0.7087 | 0.1019 | 0.1505 | 0 | 0.0097 | 0 | 0 | 0.0291 |
| SWF | 0 | 0 | 0.0714 | 0 | 0 | 0 | 0.0238 | 0.4762 | 0.4286 |
| OUTROS | 0.0294 | 0.0588 | 0.1765 | 0 | 0.0294 | 0 | 0.2353 | 0.0294 | 0.4412 |

SERVIDOR “WWW.GLOBO.COM”

| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.2138 | 0.2419 | 0.2730 | 0.0425 | 0.2273 | 0 | 0 | 0.0015 | 0 |
| HTML | 0.4256 | 0.0119 | 0.2934 | 0.0408 | 0.1601 | 0 | 0 | 0.0031 | 0.0651 |
| GIF | 0.0020 | 0.4034 | 0.0494 | 0.2572 | 0.0227 | 0.0002 | 0 | 0.0101 | 0.2551 |
| JPG | 0.0108 | 0.1028 | 0.2749 | 0.2254 | 0.2716 | 0 | 0 | 0.0007 | 0.1138 |
| JAVASCRIPT | 0.0004 | 0.0901 | 0.0545 | 0.8129 | 0.0011 | 0.0001 | 0 | 0.0032 | 0.0376 |
| CSS | 0.0040 | 0.0553 | 0.1461 | 0.0557 | 0.7095 | 0 | 0 | 0.0002 | 0.0293 |
| PNG | 0 | 0 | 0 | 0.1429 | 0 | 0.7143 | 0 | 0 | 0.1429 |
| SWF | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUTROS | 0.0149 | 0.3433 | 0.0448 | 0 | 0.0224 | 0 | 0 | 0.1119 | 0.4627 |

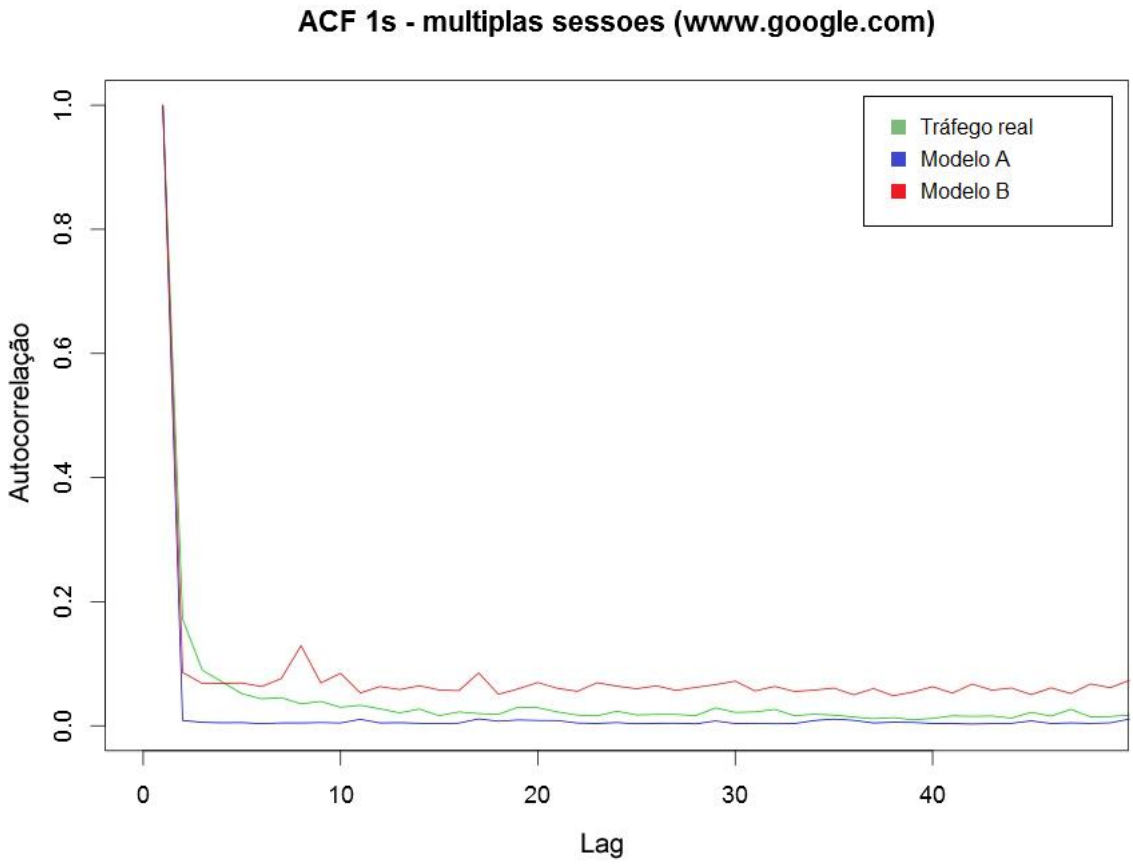
SERVIDOR “WWW.GOOGLE.COM”

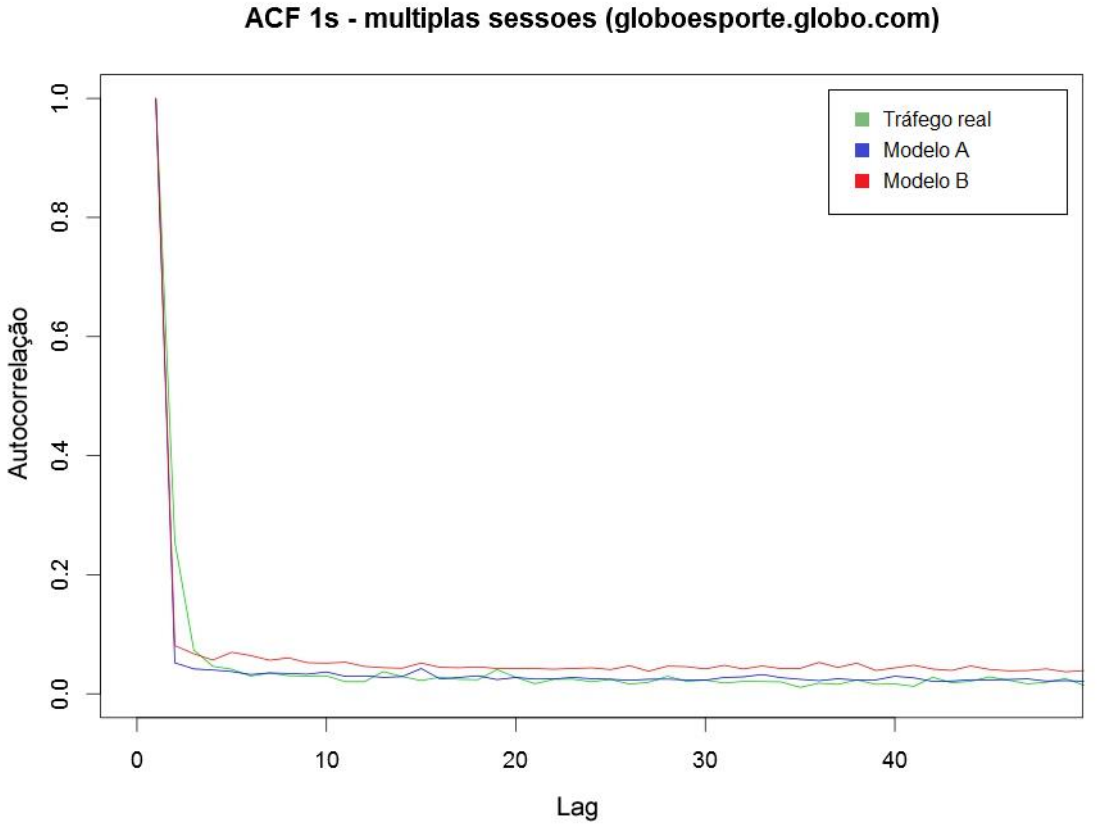
| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.4991 | 0.1166 | 0.1627 | 0.0050 | 0.0550 | 0.0341 | 0 | 0.1275 | 0 |
| HTML | 0.5130 | 0.0280 | 0.0542 | 0.0105 | 0.0065 | 0.0317 | 0 | 0.0325 | 0.3236 |
| GIF | 0.0542 | 0.5241 | 0.0504 | 0.0200 | 0.0145 | 0.0924 | 0 | 0.0052 | 0.2392 |
| JPG | 0.0800 | 0.1431 | 0.4582 | 0.0069 | 0.0814 | 0.0387 | 0 | 0.0069 | 0.1847 |
| JAVASCRIPT | 0.0841 | 0.1336 | 0.0172 | 0.3233 | 0.0022 | 0.3578 | 0 | 0.0086 | 0.0733 |
| CSS | 0.0690 | 0.0810 | 0.3762 | 0.0024 | 0.3083 | 0.0500 | 0 | 0.0036 | 0.1095 |
| PNG | 0.0558 | 0.1340 | 0.0354 | 0.0933 | 0.0214 | 0.5491 | 0 | 0.0097 | 0.1013 |
| SWF | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUTROS | 0.0543 | 0.0059 | 0.0112 | 0 | 0.0006 | 0.0012 | 0 | 0.5830 | 0.3438 |

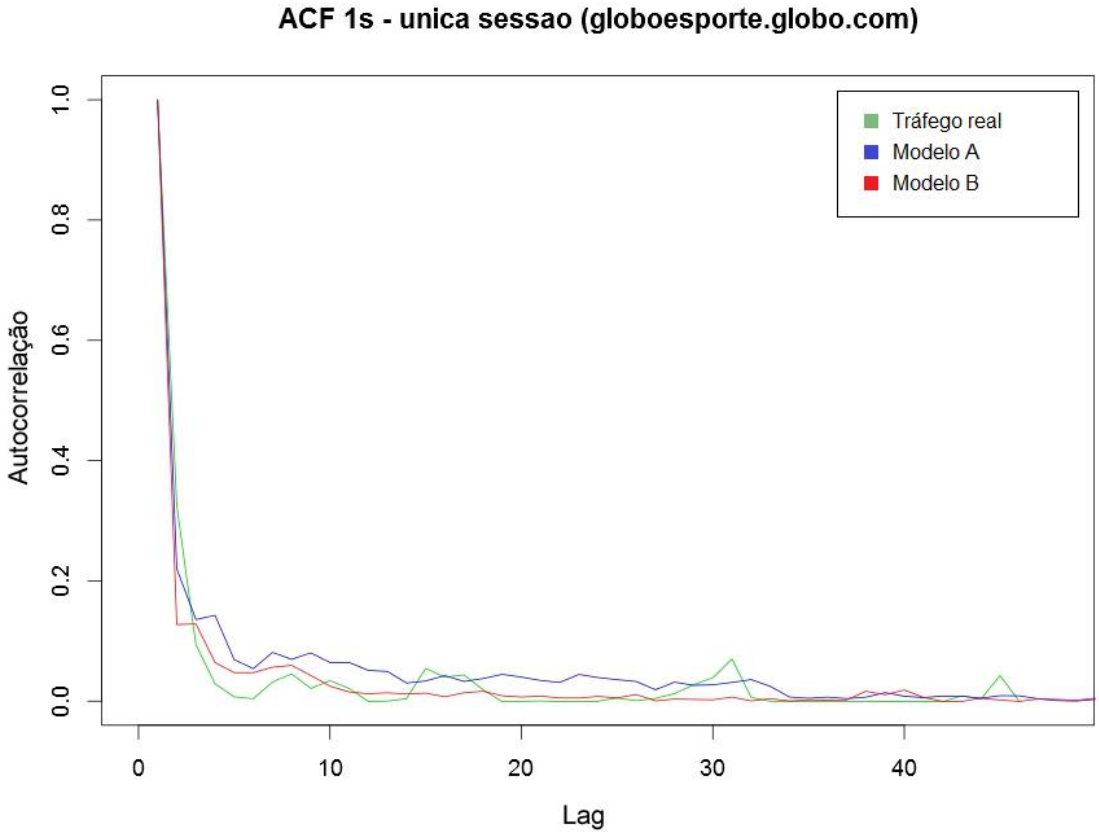
SERVIDOR “WWW.ICISAUDE.ORG.BR”

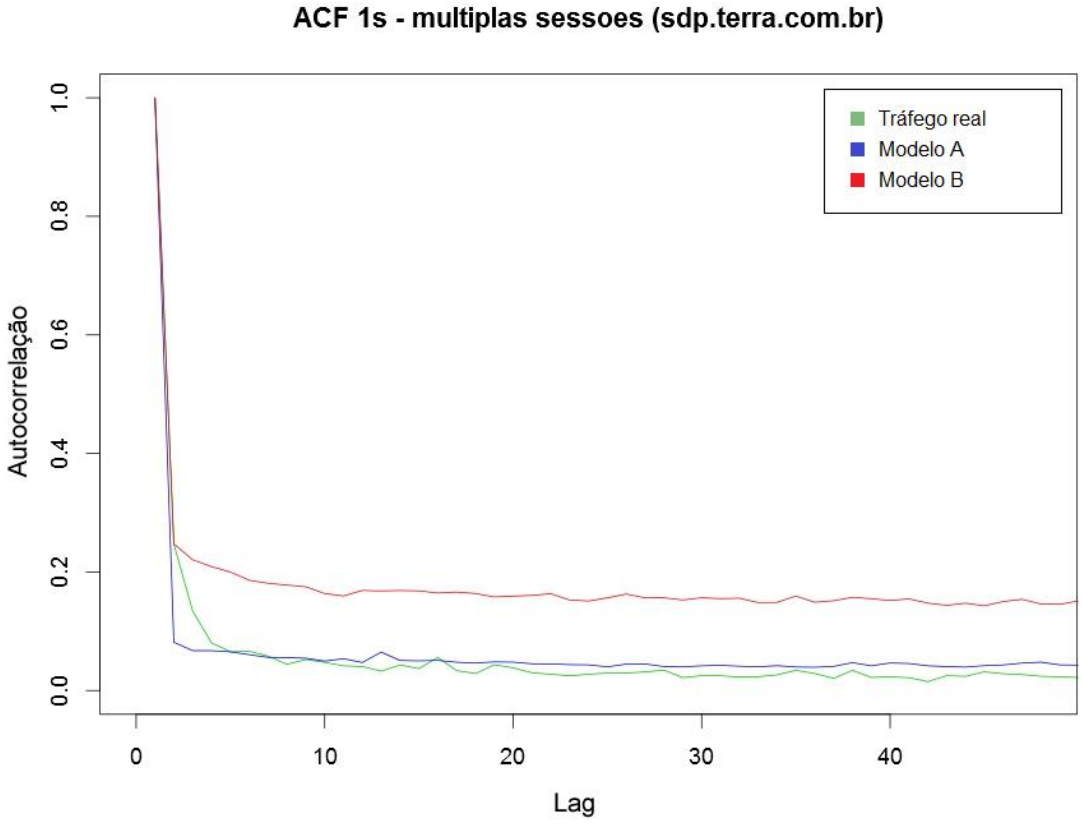
| Classe | HTML | GIF | JPG | JAVASCRIPT | CSS | PNG | SWF | OUTROS | FIM |
|---|---|---|---|---|---|---|---|---|---|
| INÍCIO | 0.0174 | 0.1136 | 0.3366 | 0 | 0.1792 | 0.0174 | 0 | 0.3360 | 0 |
| HTML | 0.0241 | 0.1556 | 0.0630 | 0 | 0.4907 | 0.0981 | 0 | 0.0056 | 0.1630 |
| GIF | 0.0172 | 0.5703 | 0.1655 | 0.0006 | 0.0837 | 0.0126 | 0 | 0.0768 | 0.0733 |
| JPG | 0.0071 | 0.2203 | 0.4489 | 0.0037 | 0.1443 | 0.0159 | 0 | 0.0834 | 0.0764 |
| JAVASCRIPT | 0.3288 | 0.3562 | 0.0137 | 0 | 0.0274 | 0.2466 | 0 | 0.0274 | 0 |
| CSS | 0.0034 | 0.1328 | 0.5970 | 0.0036 | 0.1028 | 0.0152 | 0 | 0.0429 | 0.1022 |
| PNG | 0.0813 | 0.1335 | 0.2597 | 0 | 0.1699 | 0.1553 | 0 | 0.0449 | 0.1553 |
| SWF | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| OUTROS | 0.0156 | 0.1054 | 0.1542 | 0.0001 | 0.0516 | 0.0152 | 0 | 0.5338 | 0.1240 |

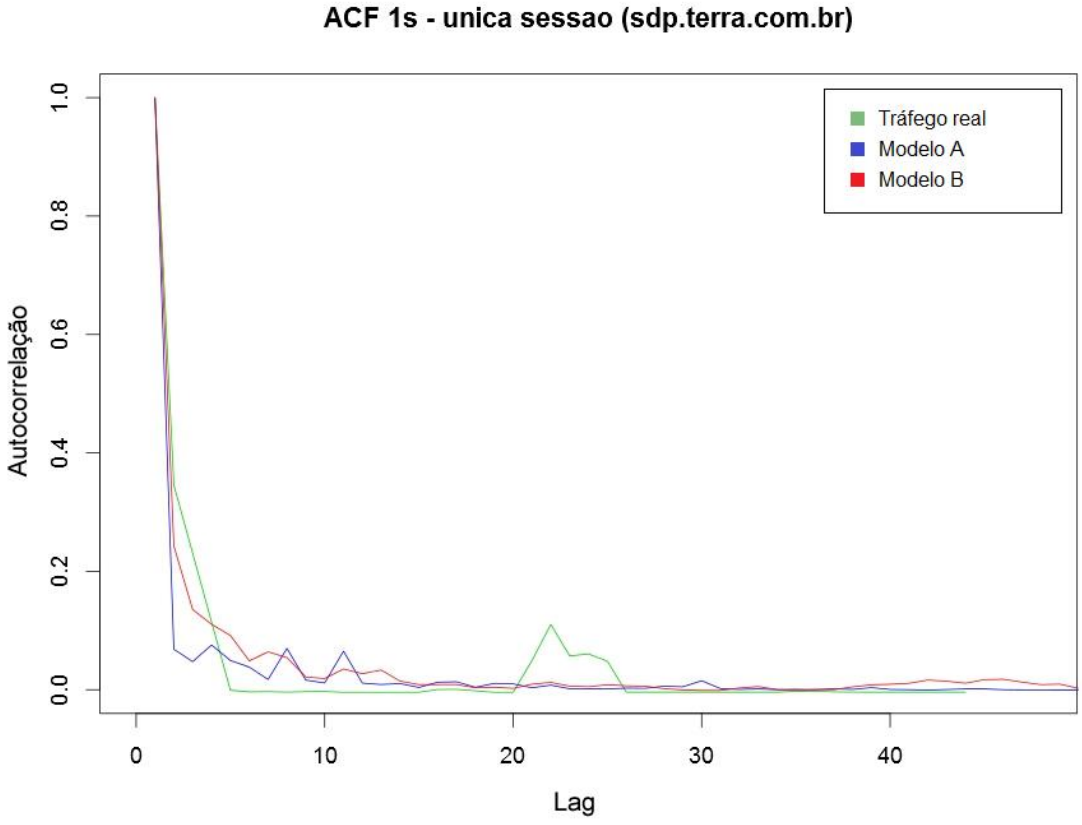
APÊNDICE L – ACF do tamanho de arquivos transferidos

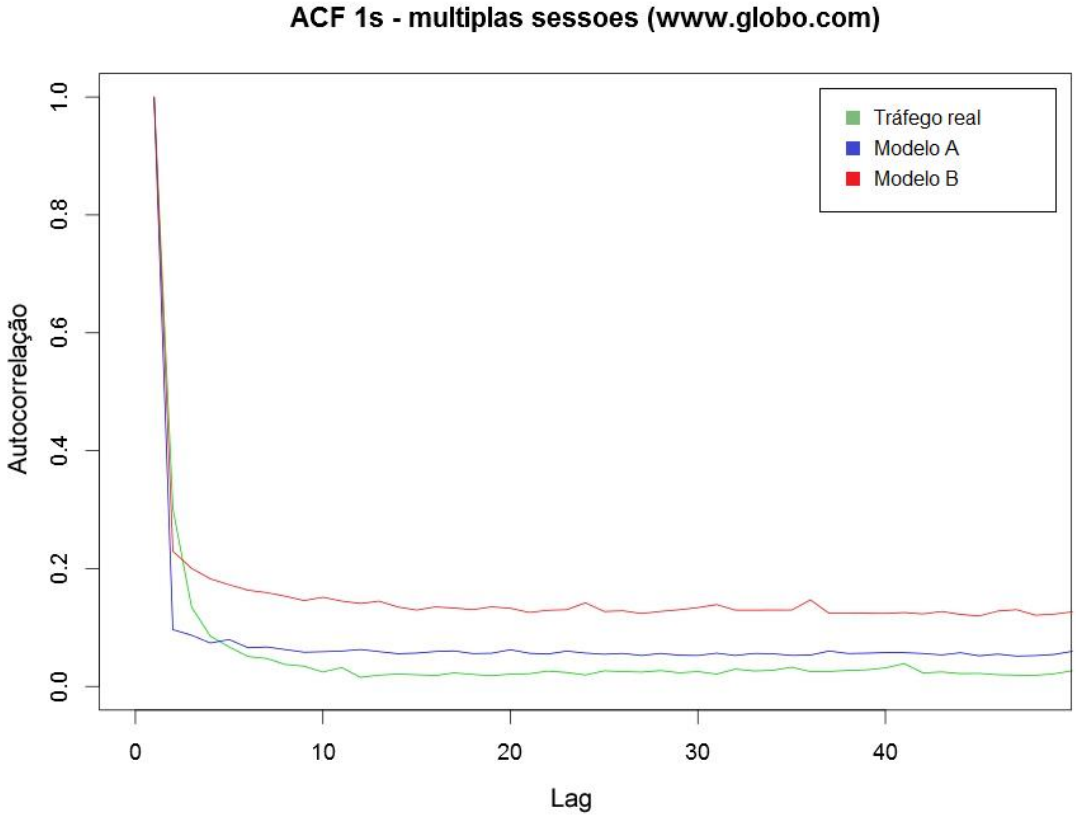


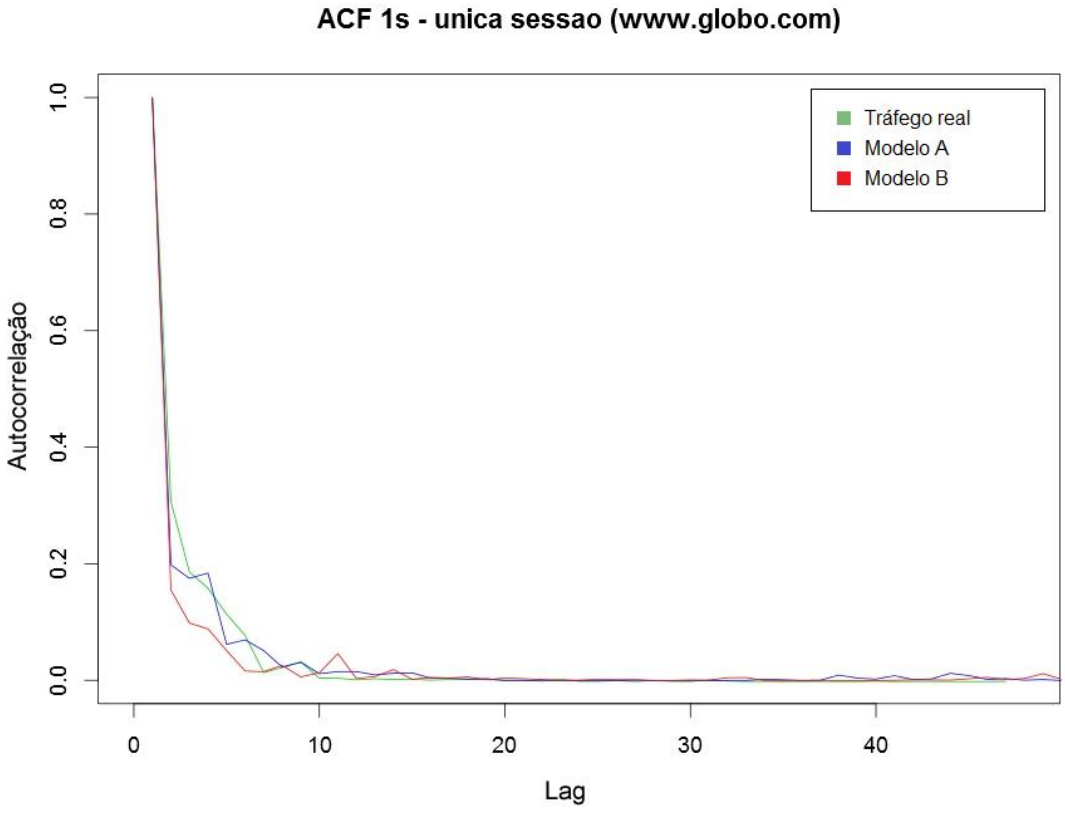

ACF 1s - multiplas sessoes (globoesporte.globo.com)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

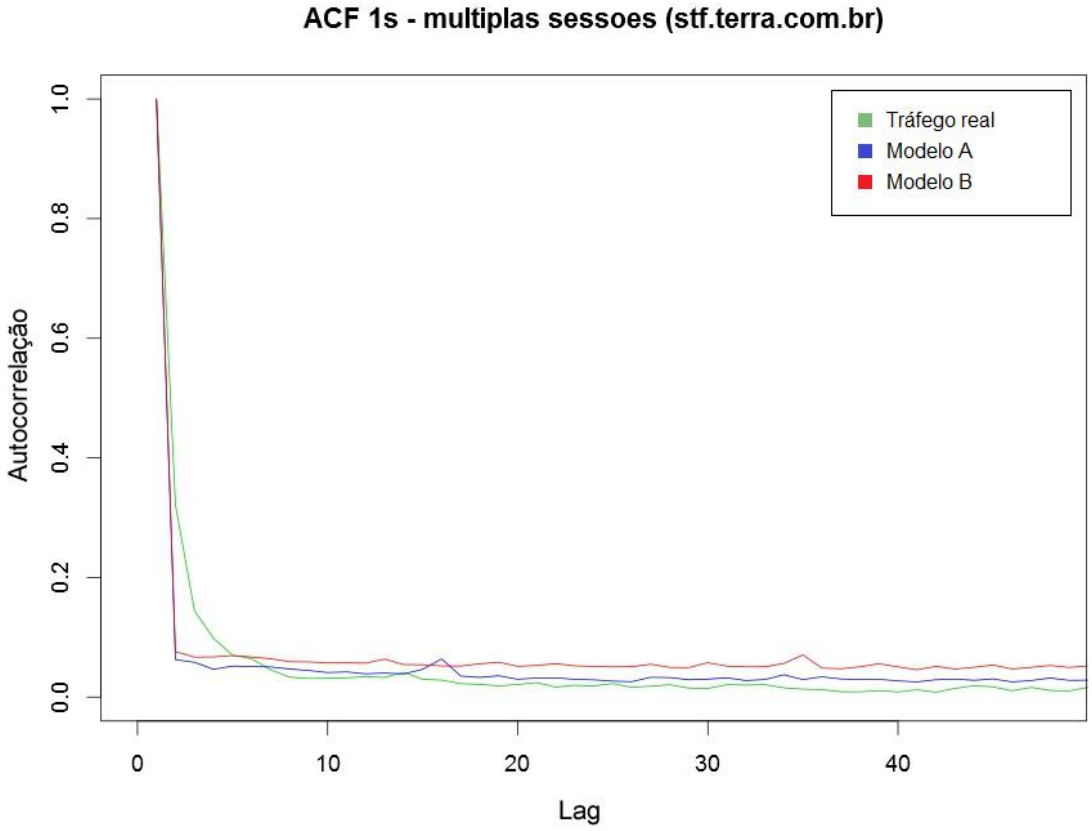

ACF 1s - unica sessao (globoesporte.globo.com)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

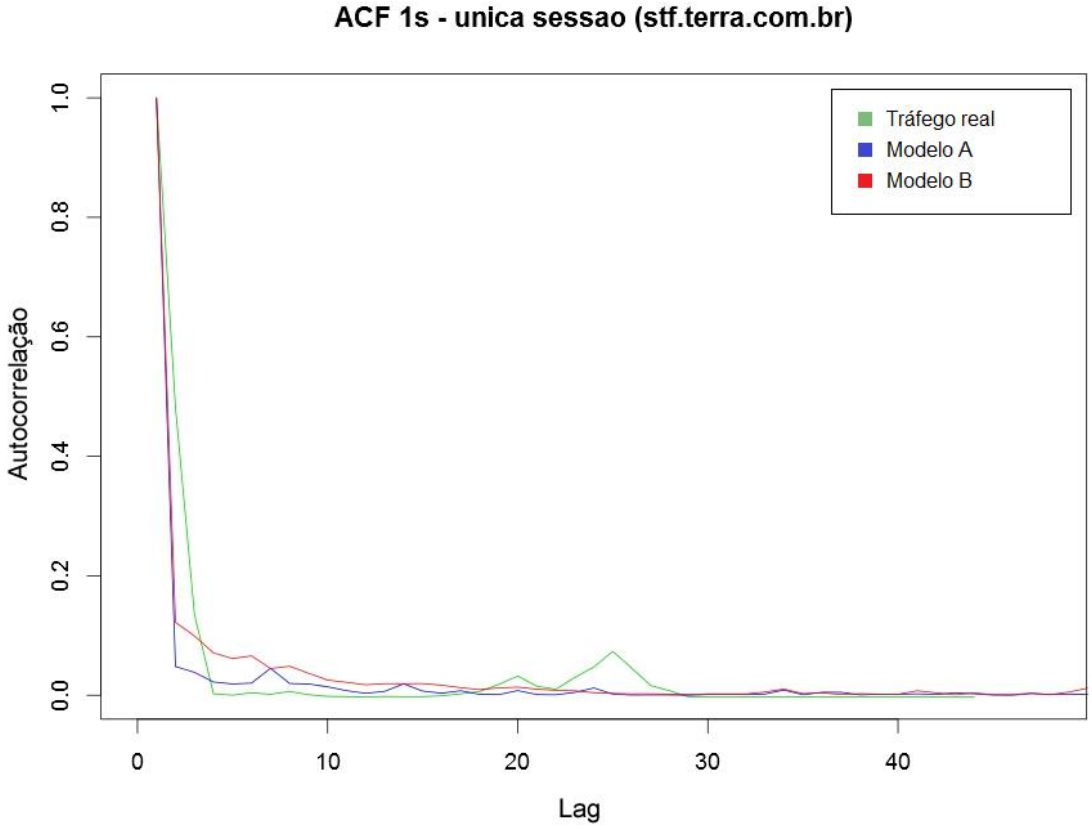

ACF 1s - multiplas sessoes (sdp.terra.com.br)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

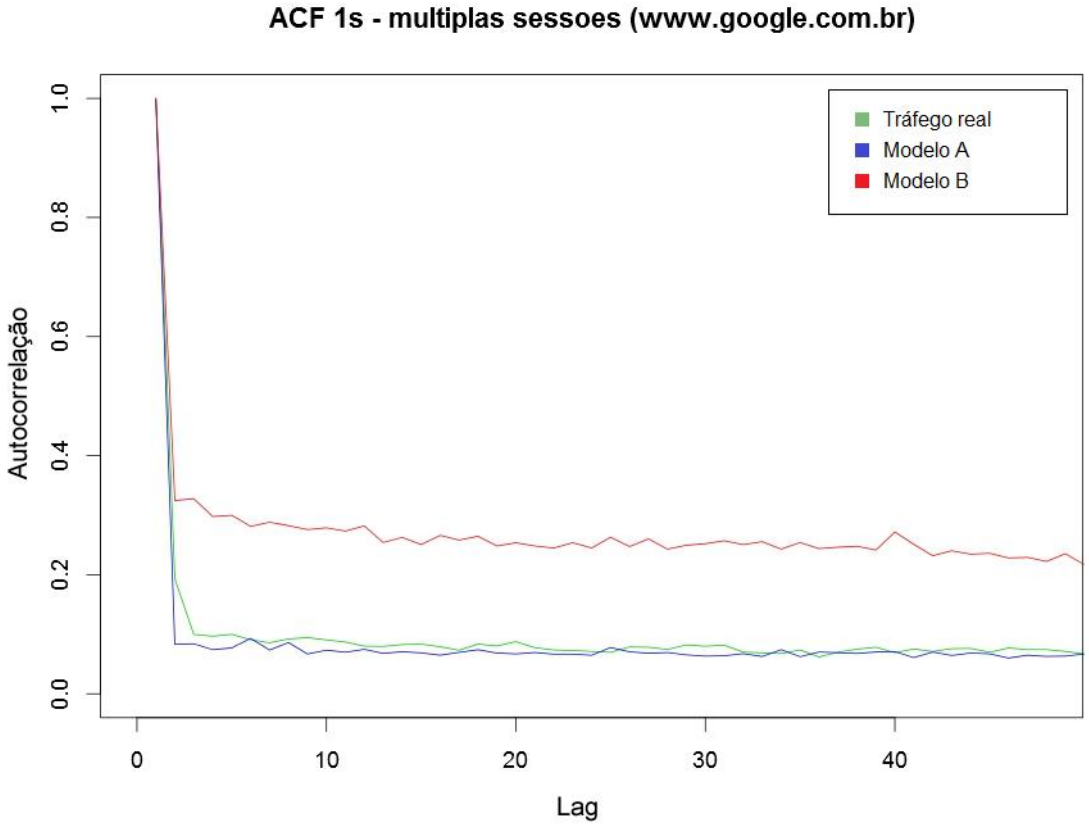

ACF 1s - unica sessao (sdp.terra.com.br)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

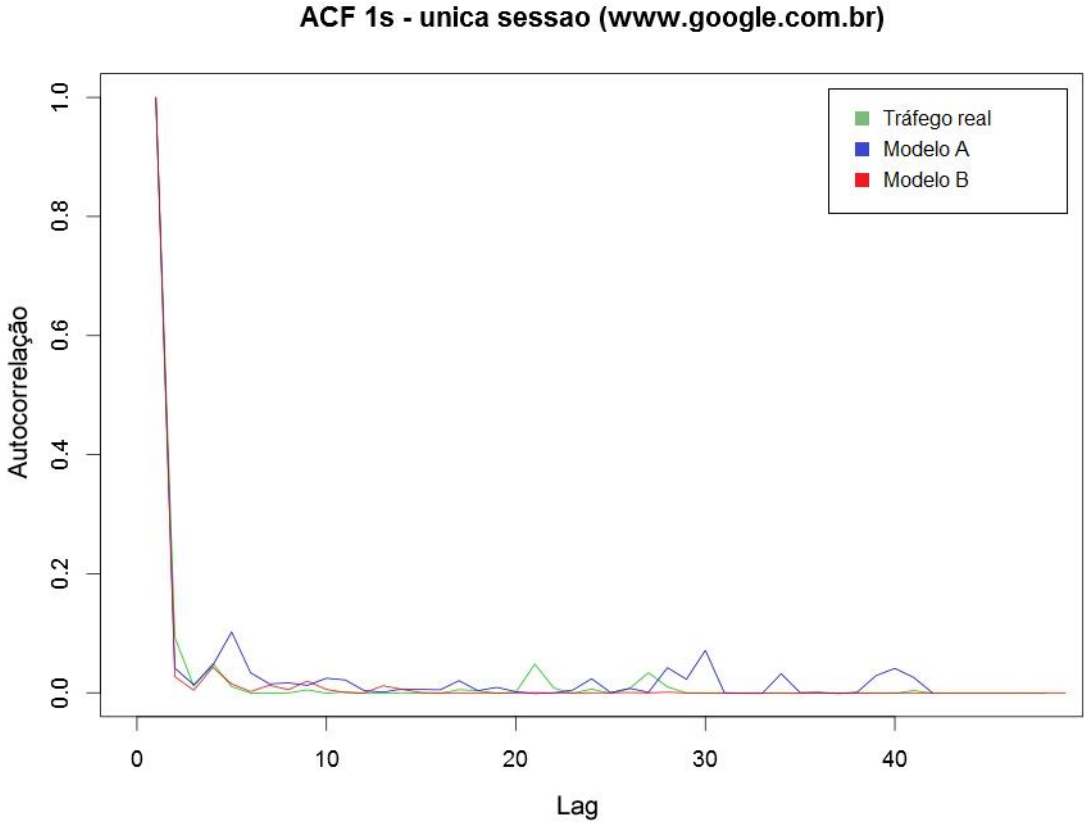

ACF 1s - multiplas sessoes (www.globo.com)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag


ACF 1s - unica sessao (www.globo.com)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag


ACF 1s - multiplas sessoes (stf.terra.com.br)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag


ACF 1s - unica sessao (stf.terra.com.br)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag


ACF 1s - multiplas sessoes (www.google.com.br)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag


ACF 1s - unica sessao (www.google.com.br)
Tráfego real
Modelo A
Modelo B
Autocorrelação
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag