
**Mestrado em Redes e Serviços de Comunicação**

*Análise e Modelização de Sistemas e Redes*

# Trabalho de análise de tráfego Internet

**Jaime Sousa Dias** **mrs01013**
**Rui Manuel Oliveira** **mrs01020**

**2003-4-22**

# ÍNDICE

## FIGURAS



## TABELAS

# 1 RESPOSTAS

## *1.1 Resposta 1*

*Calcule a percentagem de pacotes de cada um dos 3 protocolos mais frequentes no traço. Faça o mesmo para os bytes. Existe alguma diferença entre os valores obtidos. Se sim, explique a diferença.*

Com base na Tabela 1, Figura 1 e Figura 2 é possível concluír que, os três protocolos (sobre IP) mais frequentes são o TCP, o UDP e o ICMP, tanto pela quantidade de pacotes transmitidos como pela quantidade de bytes. Salienta-se, no entanto, que a % de pacotes não é igual à % de bytes por protocolo.

No caso do TCP deve-se principalmente, aos segmentos de acks puros (segmentos de confirmação sem dados) que representam uma grande % dos pacotes transmitido (ver Tabela 2), cujo tamanho é reduzido - 40bytes (20 do IP + 20 TCP). Além disso, o TCP assenta sobre mecanismos de controlo de fluxo e congestionamento, os quais tendem a levar a uma menor eficiência, i.e, a uma maior quantidade de pacotes em média mais pequenos.

O UDP, ao contrário do TCP, não assenta em nenhum mecanismo de controlo de fluxo ou de congestionamento, como tal, pelas razões inversas às apresentadas para o TCP tende a apresentar pacotes (datagramas) em média maiores do que segmentos TCP.

As mensagens ICMP são tipicamente pequenas, justificando por isso a diferença entre a quantidade de pacotes e bytes transmitidos.

**Tabela 1 Quantidade/% de pacotes e bytes dos 3 protocolos mais frequentes**

| *Protocolo* | *Nº pacotes* | *% pacotes* | *Nº bytes* | *% bytes* |
|---|---|---|---|---|
| **TCP (6)** | **114328** | **76,4%** | **59709407** | **67,4%** |
| **UDP (17)** | **34219** | **22,9%** | **28762772** | **32,5%** |
| **ICMP (1)** | **975** | **0,7%** | **90988** | **0,1%** |
| PIM (103) | 96 | 0,1% | 17000 | 0,0% |
| * (169) | 7 | 0,0% | 504 | 0,0% |
| IGMP (2) | 2 | 0,0% | 88 | 0,0% |
| *Total* | *149627* | *100,0%* | *88580759* | *100,0%* |

* de acordo com a IANA este ID ainda não foi atribuído

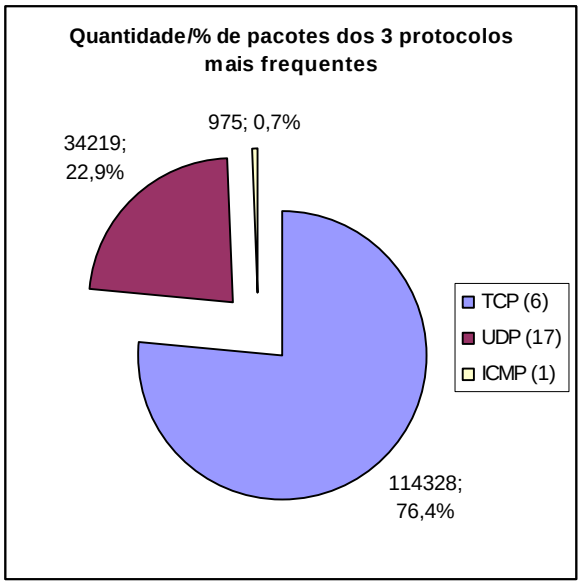


**Figura 1 Quantidade/% de pacotes dos 3 protocolos mais frequentes**

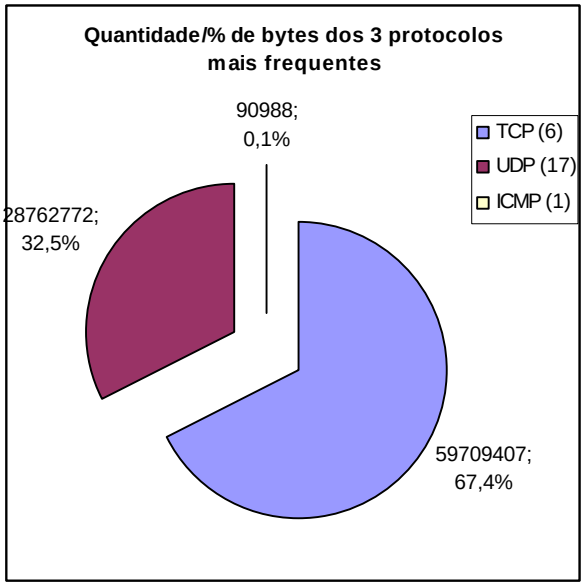


**Figura 2 Quantidade/% de bytes dos 3 protocolos mais frequentes**

## *1.2 Resposta 2*

*Calcule e represente a função distribuição de comprimento dos pacotes. Comente a forma e os valores da curva.*

Como pode ser constatado nas figuras que se seguem, a probabilidade de encontramos um pacote com um dado tamanho não é sempre a mesma. Com efeito, verifica-se uma grande quantidade de pacotes com tamanho com tamanho 40 bytes (25%) e com tamanho próximo dos 1500 bytes (20%). Para além destes duas referências, verificam-se pequenas excepções de pequenas quantidades de pacotes com tamanhos 774 e 1216 bytes e uma "maior inclinação" (ver Figura 4 e Figura 6) entre os 40 os 100 bytes, podendo-se afirmar que a

probabilidade de encontrarmos pacotes com um dado tamanho entre os 40 e aproximadamente 1500 bytes é constante.

A justificação para a grande quantidade de pacotes de tamanho 40 bytes, são os acks puros TCP (20 IP +20 TCP). Enquanto que para os 1500 bytes é devido ao MTU das redes ethernet típicas (10/100 Mbit/s), que obriga a que os datagranas IP sejam fragmentados em pacotes com tamanho máximo igual ao MTU.

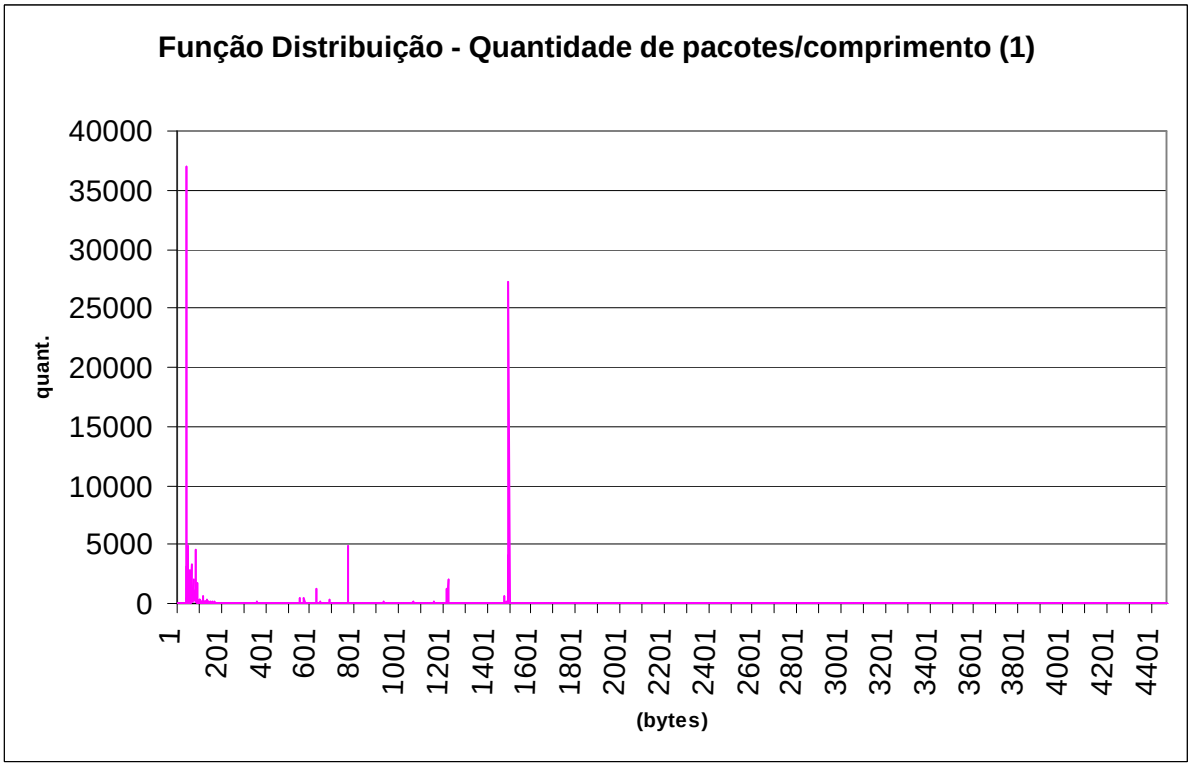


**Figura 3 Função distribuição -Quantidade de pacotes por comprimento**

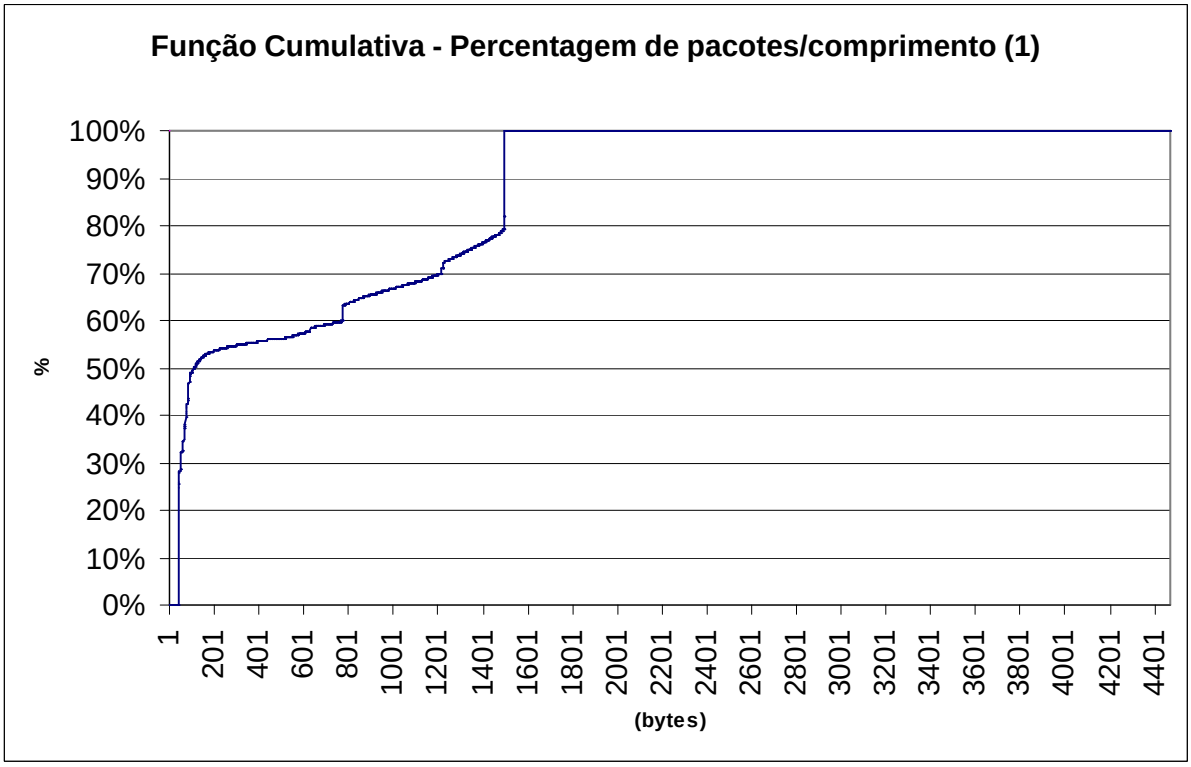


**Figura 4 Função cumulativa - % de pacotes por comprimento**

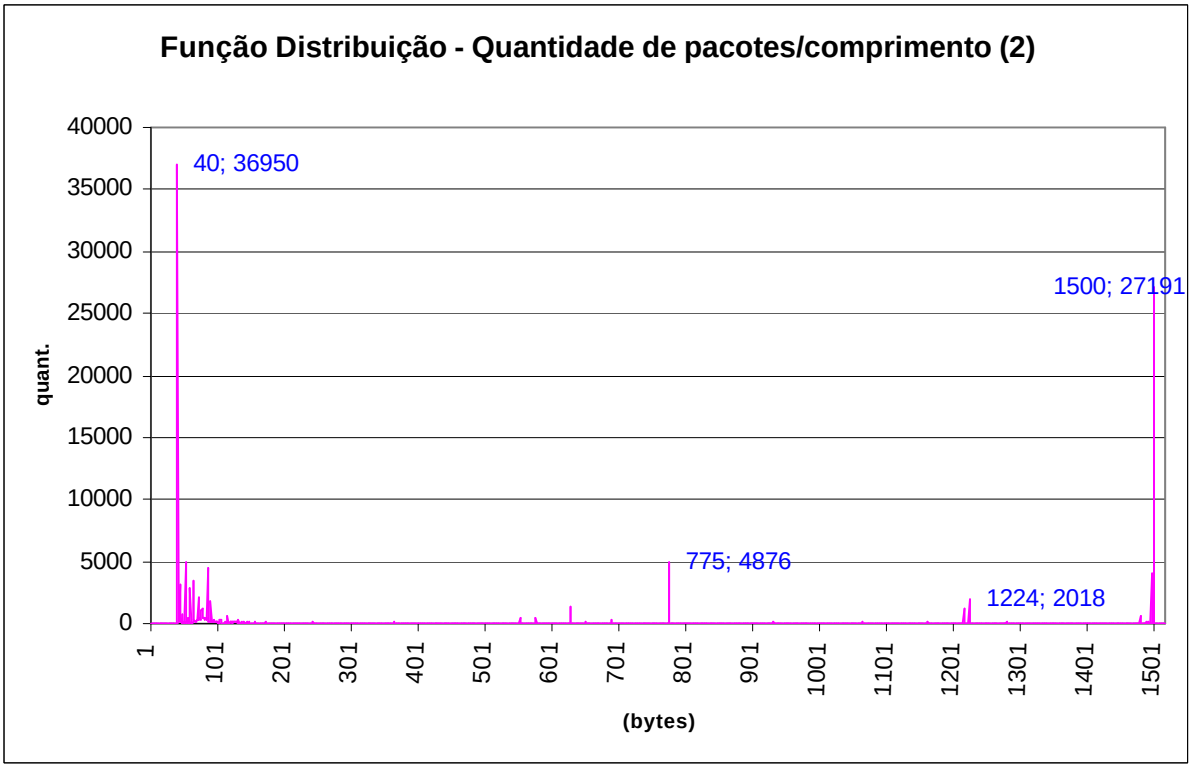


**Figura 5 Função distribuição -Quantidade de pacotes por comprimento (comprimento <=1500bytes)**

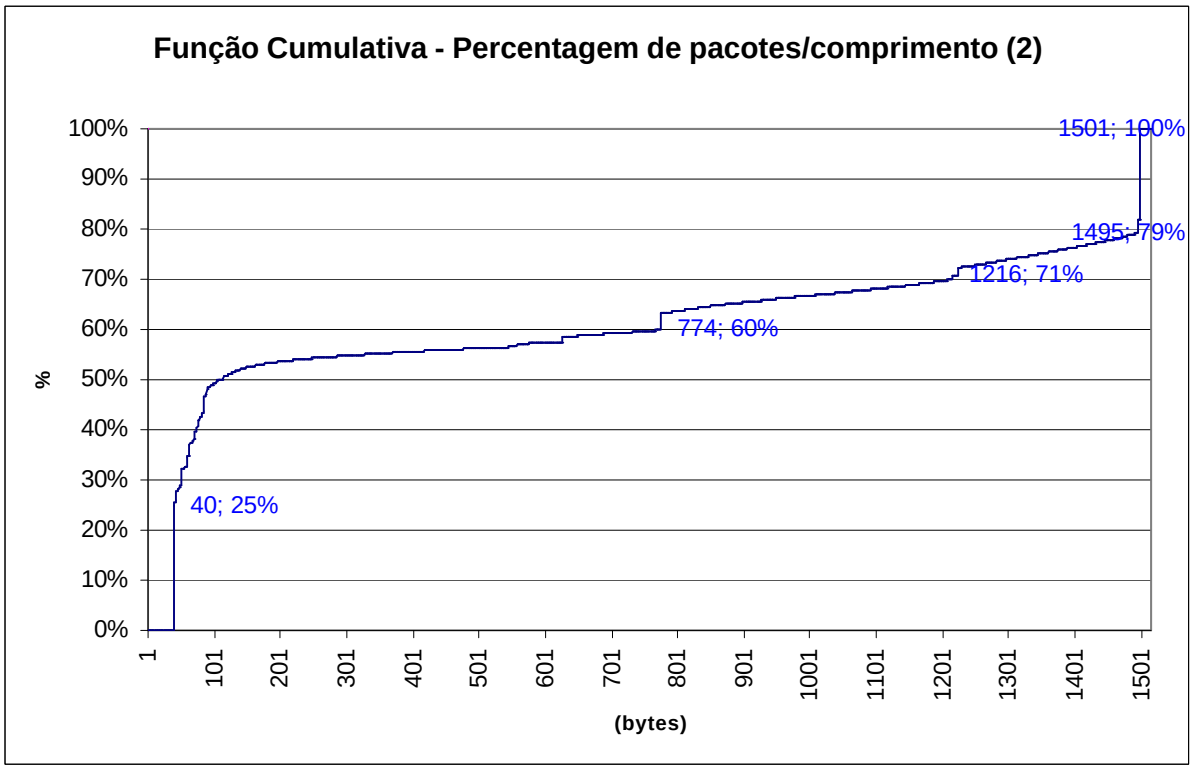


**Figura 6 Função cumulativa - % de pacotes por comprimento (comprimento <=1500bytes)**

## *1.3 Resposta 3*

*Calcule a percentagem de pacotes SMTP, HTTP, FTP, NTP e DNS. Calcule ainda a percentagem de pacotes que nao usam uma "porta-bem-conhecida". Calcule a percentagem de pacotes que sao ACKs puros (sem dados)*

Em http://www.rfc-editor.org/rfc/rfc1700.txt está definido que "porta-bem-conhecida" é toda a porta abaixo de 1024:

*"WELL KNOWN PORT NUMBERS*

*The Well Known Ports are controlled and assigned by the IANA and on most systems can only be sed by system (or root) processes or by programs executed by privileged users. (...) The assigned ports use a small portion of the possible port numbers. For many years the assigned ports were in the range 0-255. Recently, the range for assigned ports managed by the IANA has been expanded to the range 0-1023."*

Para cada porto consideramos todo o pacote que tenha um dos portos igual ao pretendido. Pacotes sem porta-bem conhecida são todos aqueles cujos portos são ambos superiores a 1023.

**Tabela 2 Quantidade de pacotes por porta**

| *Protocolo* | *pacotes* | *%* |
|---|---|---|
| **FTP-Data (20)** | 17576 | 11,75% |
| **FTP (21)** | 26 | 0,02% |
| **SMTP (25)** | 1488 | 0,99% |

| | | |
|---|---|---|
| **DNS (53)** | 823 | 0,55% |
| **HTTP (80)** | 28686 | 19,17% |
| **NNTP (119)** | 2170 | 1,45% |
| **NTP (123)** | 545 | 0,36% |
| **outros <=1023** | 15444 | 10,32% |
| **>1023** | 82869 | 55,38% |
| *Total* | *149627* | *100,00%* |

| | | |
|---|---|---|
| **Acks TCP puros** | 33613 | 22,46% |

Dos vários protocolos com portos bem conhecidos, em Tabela 2 apresentamos alguns. É possível observar uma natural maior % de pacotes que suportam o HTTP (19,17%), seguida das ligações de dados FTP (modo passivo). Verifica-se também que existe uma maior % de tráfego que não utiliza portos bem conhecidos. Neste tráfego incluem-se os protocolos não registados no IANA como exemplo os protocolos das aplicações P2P, hoje muito difundidas.

Verificou-se que 22,46% dos pacotes analisados são ack TCP puros. Convém salientar que parte destes acks puros já estão incluídos nos pacotes seleccionados para os protocolos acima referidos.

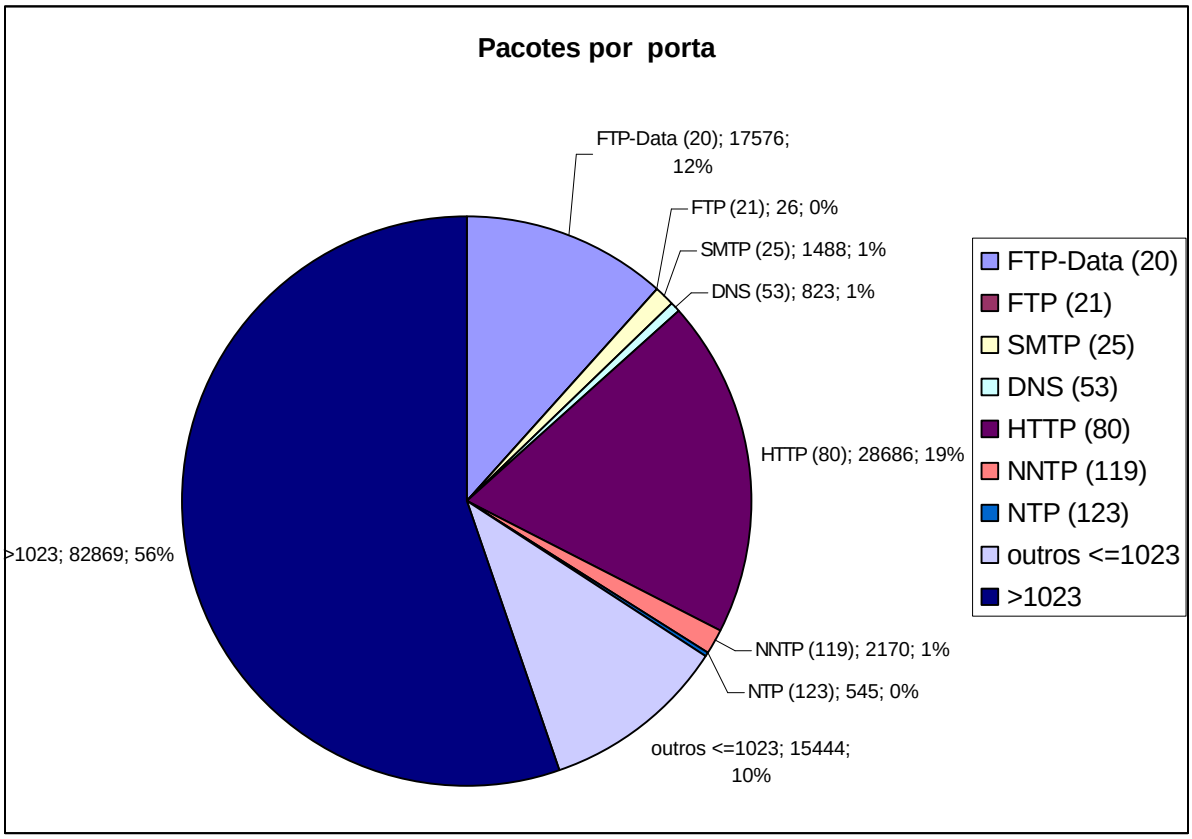


**Figura 7 Quantidade de pacotes por porta**

## *1.4 Resposta 4*

*Assuma que um fluxo de pacotes é uma sequencia de pacotes com o mesmo endereço de origem, endereço de destino, porta de origem, porta de destino e número de protocolo. Assuma que um fluxo deve ter pelo menos 3 pacotes. Meça o número de fluxos no traço. Quantos usam o protocolo TCP? E o protocolo UDP?*

Como pode ser observado na tabela e figura que se seguem, a maioria dos fluxos que se seguem são de tráfego TCP (96,54%), só são considerados os fluxos de protocolos de transporte, pois são os únicos com conceito de portos.

Convém salientar que de acordo com o enunciado, é assumido que cada fluxo é unidireccional, sendo que uma conexão TCP é composta por dois fluxos. Se assim não fosse (uma conexão TCP considerada apenas um fluxo), o nº de fluxos TCP diminuia para cerca de metade e representaria cerca de 93% dos fluxos.

**Tabela 3 Quantidade de fluxos por protocolo da camada de transporte**

| | | |
|---|---|---|
| **Fluxos TCP** | 2680 | 96,54% |
| **Fluxos UDP** | 96 | 3,46% |
| *Total de fluxos* | *2776* | *100,00%* |

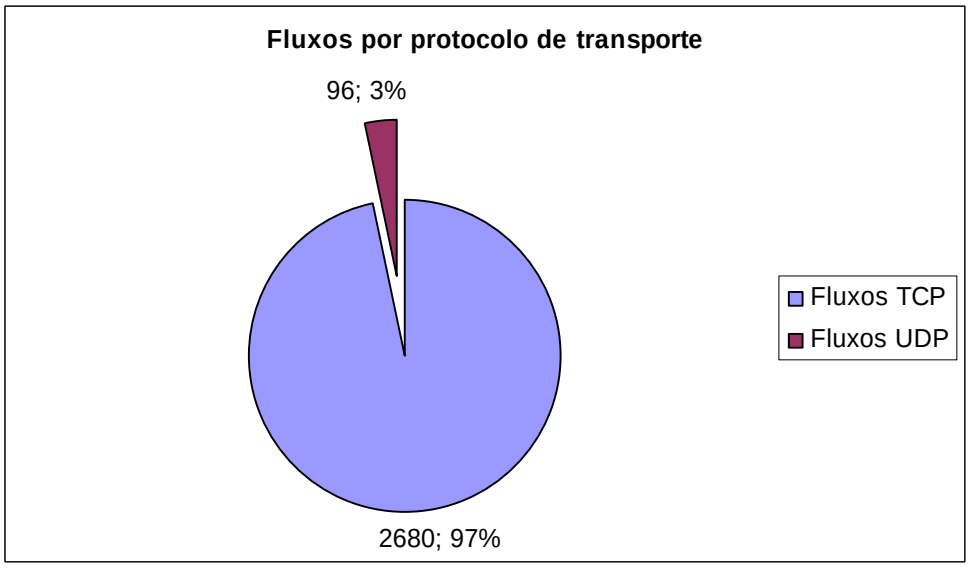


**Figura 8 Quantidade de fluxos por protocolo da camada de transporte**

## *1.5 Resposta 5*

*Construa o seguinte diagrama. No eixo do xx representamos o número de ordem de cada fluxo por ordem decrescente de bytes transferidos. Para cada X, o seu valor no eixo dos yy deve representar a percentagem de bytes de todos os fluxos com ordem menor ou igual a X.*

*O que observa? De quantos fluxos precisa um router de se recordar para manter registo de pelo menos 90% do tráfego.*

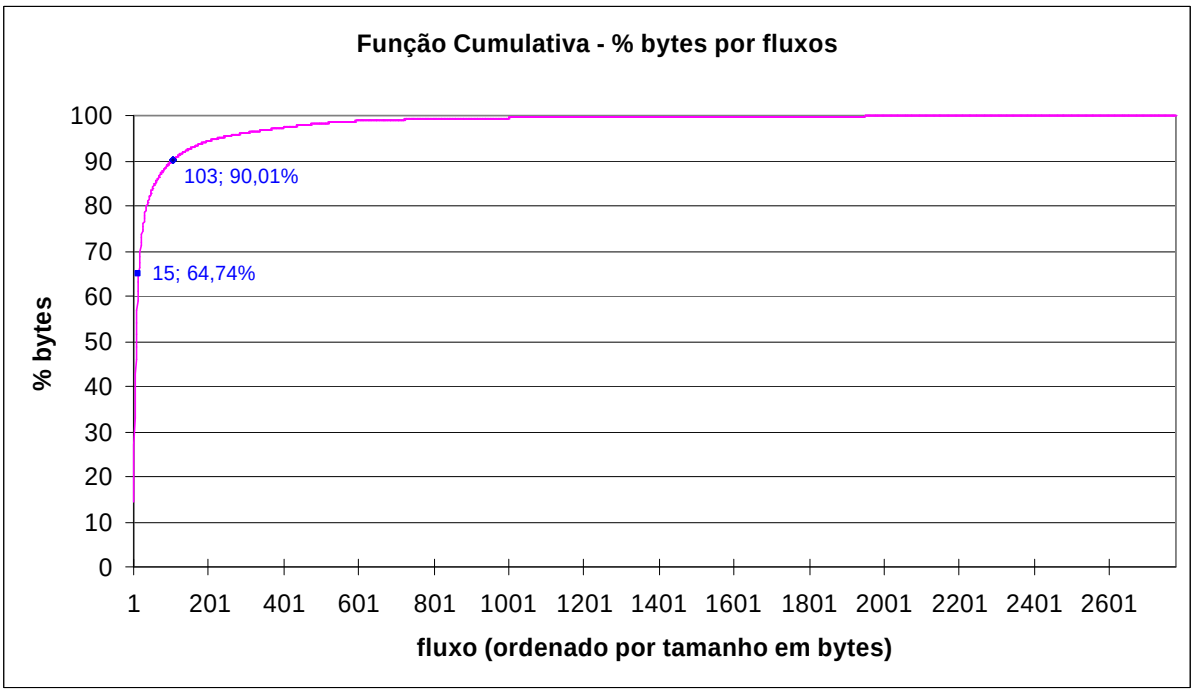


**Figura 9 Função Cumulativa - % de bytes por fluxos**

Verifica-se pela Figura 9 que se um router conseguir memorizar apenas os fluxos maiores, então necessita de manter em memória cerca de 100 fluxos (103 neste caso) para manter o registo de 90% do tráfego. Sendo que neste caso, 103 fluxos representam apenas 3,7% do nº total de fluxos.

## *1.6 Resposta 6*

*Sobre o mesmo gráfico, represente o débito do traço como uma função do tempo (relacione o tempo com os fluxos representados da forma que julgar mais conveniente). Use 3 curvas distintas, uma cada intervalo de contagem do debito - 10 ms, 100 ms e 1 s.*

Nas figuras que se seguem são apresentados três gráficos com o débito do traço em função do tempo para três períodos de amostragem (Ts), 1s, 100ms e 10ms respectivamente (Figura 10, Figura 12 e Figura 13).

É também apresentado um gáfico com os débitos dos 15 maiores fluxos (responsáveis por 64% do tráfego) e o total dos restantes em função do tempo com Ts=1s (Figura 11). Não são apresentados mais gráficos deste género para os restantes Ts porque a conclusão quanto à variação de Ts obtem-se com base nos 3 gráficos acima referidos.

É importante referir que nesta questão não é feita diferenciação dos fluxos quanto ao seu sentido (interface), por isso o limite la ligação OC-3 = 155,52 Mbit/s é ultrapassado.

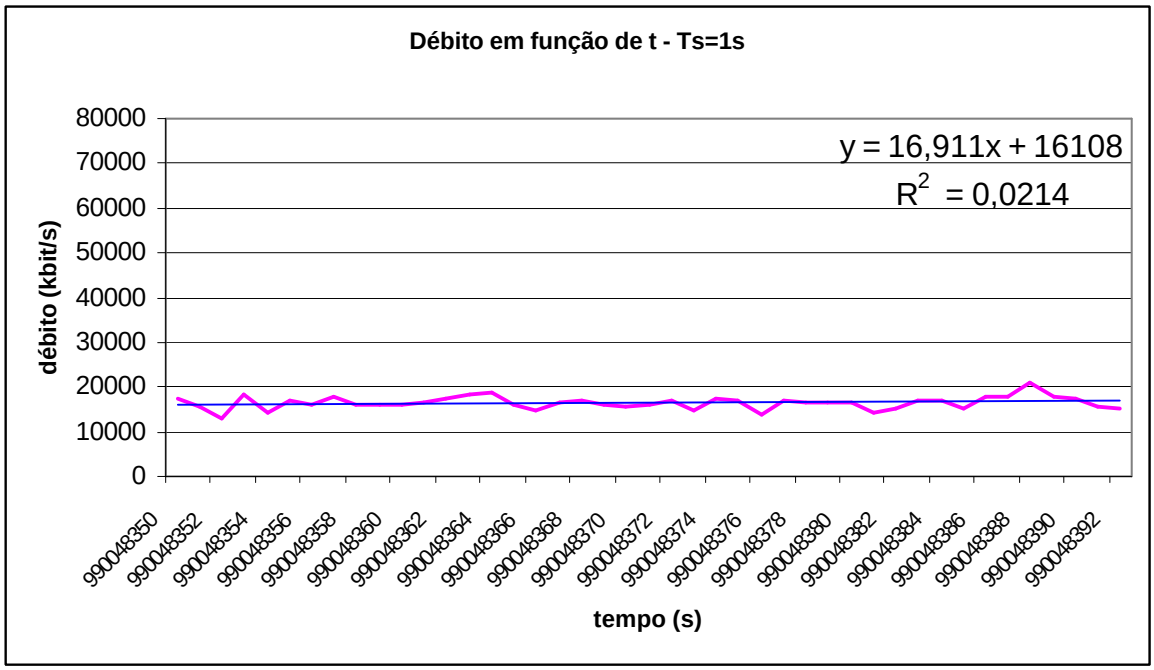


**Figura 10 Débito em função do tempo, para Ts=1s**

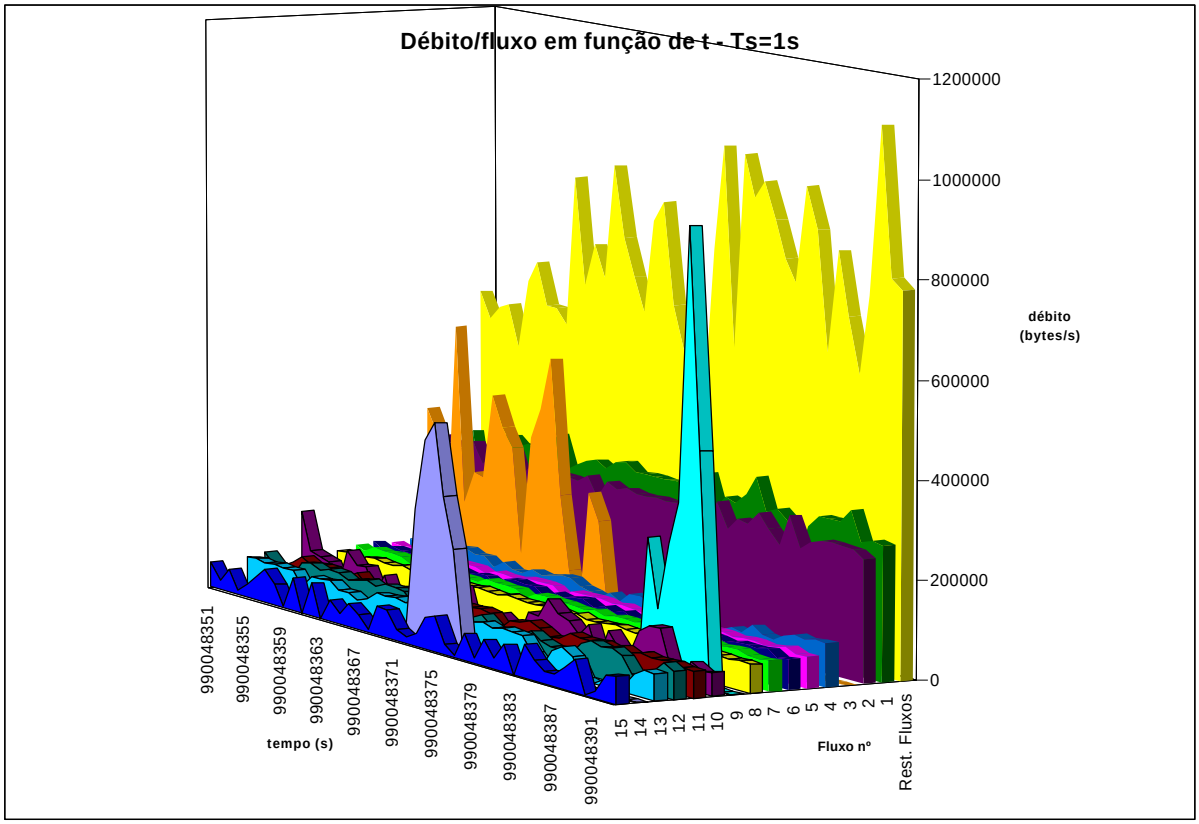


**Figura 11 Débito/fluxo em função do tempo, para Ts=1s**

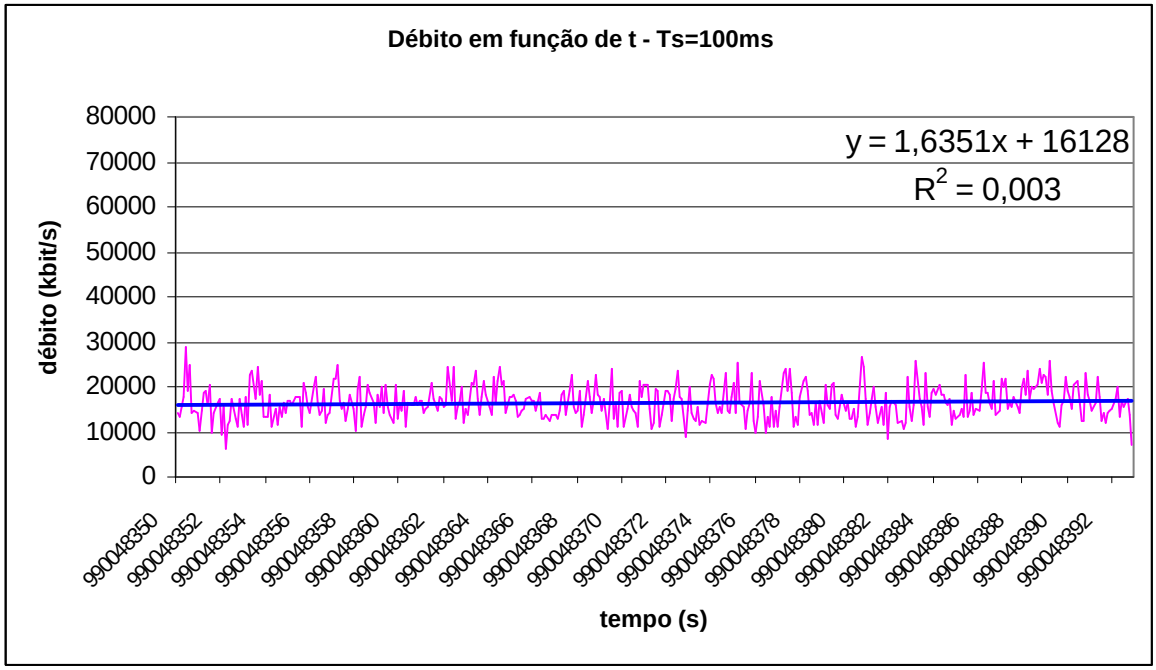


**Figura 12 Débito em função do tempo, para Ts=100ms**

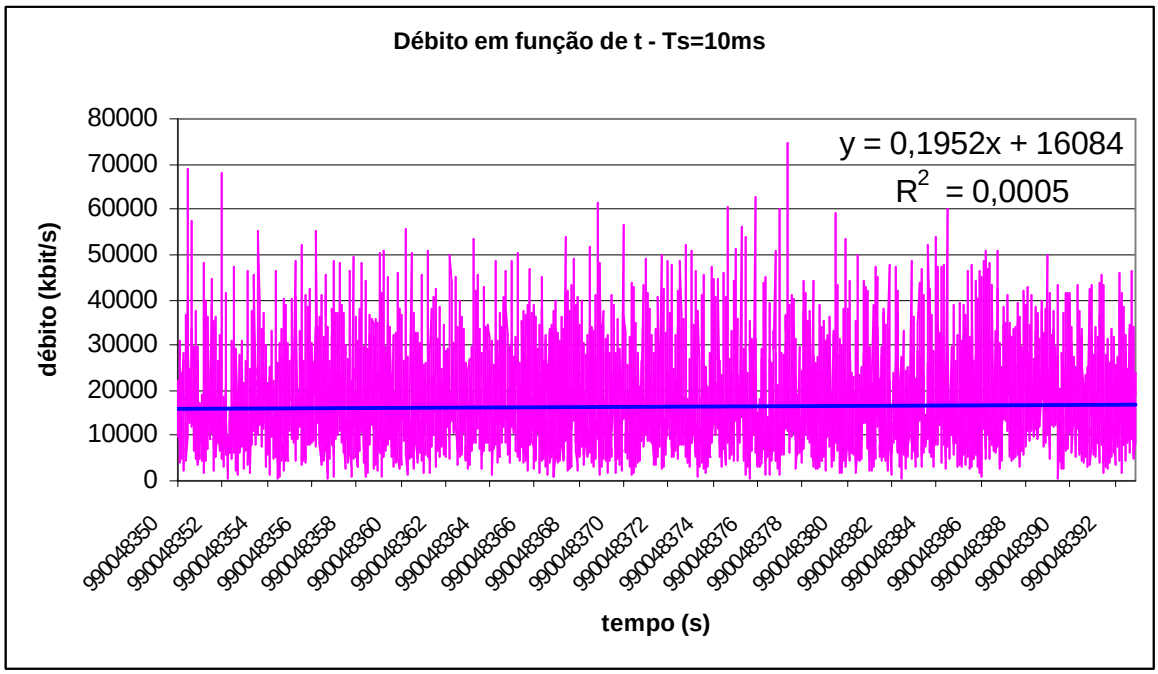


**Figura 13 Débito em função do tempo, para Ts=10ms**

No que respeita à variação de Ts, verifica-se que quanto maior este for menor é o valor de pico e mais próximo está o débito de cada amostra do débito médio. Para além da visualização gráfica, pode ser comprovado analiticamente, recorendo a uma regressão linear. Uma maior variação do débito (Ts menor) conduz a um $R^2$ menor, mais distante de 1. O débito médio deste traço é de cerca de 16,1 Mbit/s.

Quanto ao débito por fluxo (Figura 11) verifica-se que os fluxos com maior duração tendem a manter uma débito estabilizado, os fluxos que iniciam a meio do traço, apresentam a natural

característica de “slow start” tomando conta por instantes de uma boa parte da largura de banda, interferindo com o débito dos restantes fluxos.

## *1.7 Resposta 7*

*Isole no traço todos os pacotes da ligação HTTP que começa no tempo 990048367.932311. Inclua os pacotes dos 2 sentidos (cliente-servidor e servidor-cliente).*

*Represente os números de sequência dos dados transferidos do servidor para o cliente, como uma função tempo. O que observa? Explique.*

```
2 990048367.932311 12517377 720908 6 48 49180 80 02 2439902516 0 32768
1 990048367.934002 720908 12517377 6 48 80 49180 12 3652394470 2439902517 32120
2 990048368.063948 12517377 720908 6 40 49180 80 10 2439902517 3652394471 32768
2 990048368.176132 12517377 720908 6 437 49180 80 18 2439902517 3652394471 32768
1 990048368.178181 720908 12517377 6 40 80 49180 10 3652394471 2439902914 31723
1 990048368.181335 720908 12517377 6 1500 80 49180 18 3652394471 2439902914 32120
1 990048368.181496 720908 12517377 6 1500 80 49180 18 3652395931 2439902914 32120
2 990048368.302169 12517377 720908 6 40 49180 80 10 2439902914 3652395931 32768
1 990048368.305346 720908 12517377 6 1500 80 49180 18 3652397391 2439902914 32120
1 990048368.305671 720908 12517377 6 1500 80 49180 18 3652398851 2439902914 32120
2 990048368.350959 12517377 720908 6 40 49180 80 10 2439902914 3652397391 32768
1 990048368.354674 720908 12517377 6 1500 80 49180 18 3652400311 2439902914 32120
1 990048368.354822 720908 12517377 6 1109 80 49180 18 3652401771 2439902914 32120
2 990048368.430491 12517377 720908 6 40 49180 80 10 2439902914 3652400311 32768
2 990048368.506135 12517377 720908 6 431 49180 80 18 2439902914 3652402840 32768
1 990048368.530615 720908 12517377 6 1500 80 49180 18 3652402840 2439903305 32120
1 990048368.530740 720908 12517377 6 765 80 49180 18 3652404300 2439903305 32120
2 990048368.698898 12517377 720908 6 40 49180 80 10 2439903305 3652405025 32768
2 990048368.704751 12517377 720908 6 434 49180 80 18 2439903305 3652405025 32768
1 990048368.711174 720908 12517377 6 1500 80 49180 18 3652405025 2439903699 32120
1 990048368.711264 720908 12517377 6 416 80 49180 18 3652406485 2439903699 32120
2 990048368.832495 12517377 720908 6 40 49180 80 10 2439903699 3652406485 32768
2 990048368.850317 12517377 720908 6 434 49180 80 18 2439903699 3652406861 32768
1 990048368.856396 720908 12517377 6 1500 80 49180 18 3652406861 2439904093 32120
1 990048368.856690 720908 12517377 6 1500 80 49180 18 3652408321 2439904093 32120
1 990048368.856792 720908 12517377 6 1500 80 49180 18 3652409781 2439904093 32120
1 990048368.856999 720908 12517377 6 1500 80 49180 18 3652411241 2439904093 32120
1 990048368.857112 720908 12517377 6 1500 80 49180 18 3652412701 2439904093 32120
1 990048368.857390 720908 12517377 6 1500 80 49180 18 3652415621 2439904093 32120
1 990048368.857511 720908 12517377 6 1500 80 49180 18 3652417081 2439904093 32120
1 990048368.857636 720908 12517377 6 1500 80 49180 18 3652418541 2439904093 32120
2 990048368.978299 12517377 720908 6 40 49180 80 10 2439904093 3652409781 32768
1 990048368.981508 720908 12517377 6 1500 80 49180 18 3652420001 2439904093 32120
1 990048368.981833 720908 12517377 6 1500 80 49180 18 3652421461 2439904093 32120
1 990048368.981975 720908 12517377 6 1500 80 49180 18 3652422921 2439904093 32120
2 990048368.984740 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048368.984752 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048368.984780 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048368.984786 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
1 990048368.988229 720908 12517377 6 1500 80 49180 18 3652424381 2439904093 32120
1 990048368.988405 720908 12517377 6 1500 80 49180 18 3652425841 2439904093 32120
1 990048368.988518 720908 12517377 6 1500 80 49180 18 3652427301 2439904093 32120
1 990048368.988722 720908 12517377 6 1500 80 49180 18 3652428761 2439904093 32120
1 990048368.988855 720908 12517377 6 1500 80 49180 18 3652414161 2439904093 32120
2 990048369.105050 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048369.106304 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048369.106352 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048369.114212 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048369.114217 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048369.114223 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048369.114302 12517377 720908 6 40 49180 80 10 2439904093 3652414161 32768
2 990048369.114404 12517377 720908 6 40 49180 80 10 2439904093 3652430221 32768
```

```
1 990048369.117500 720908 12517377 6 1500 80 49180 18 3652430221 2439904093 32120
1 990048369.117741 720908 12517377 6 1500 80 49180 18 3652431681 2439904093 32120
1 990048369.117897 720908 12517377 6 1500 80 49180 18 3652433141 2439904093 32120
1 990048369.118032 720908 12517377 6 1500 80 49180 18 3652434601 2439904093 32120
2 990048369.287980 12517377 720908 6 40 49180 80 10 2439904093 3652436061 32768
1 990048369.291389 720908 12517377 6 1500 80 49180 18 3652437521 2439904093 32120
1 990048369.291479 720908 12517377 6 1500 80 49180 18 3652438981 2439904093 32120
1 990048369.291674 720908 12517377 6 1500 80 49180 18 3652440441 2439904093 32120
1 990048369.291765 720908 12517377 6 532 80 49180 18 3652441901 2439904093 32120
2 990048369.412466 12517377 720908 6 40 49180 80 10 2439904093 3652436061 32768
2 990048369.414422 12517377 720908 6 40 49180 80 10 2439904093 3652436061 32768
2 990048369.414488 12517377 720908 6 40 49180 80 10 2439904093 3652436061 32768
2 990048369.414507 12517377 720908 6 40 49180 80 10 2439904093 3652436061 32768
1 990048369.839384 720908 12517377 6 1500 80 49180 18 3652436061 2439904093 32120
2 990048369.959320 12517377 720908 6 40 49180 80 10 2439904093 3652442393 32768
1 990048385.717669 720908 12517377 6 40 80 49180 11 3652442393 2439904093 32120
2 990048385.835470 12517377 720908 6 40 49180 80 10 2439904093 3652442394 32768
```

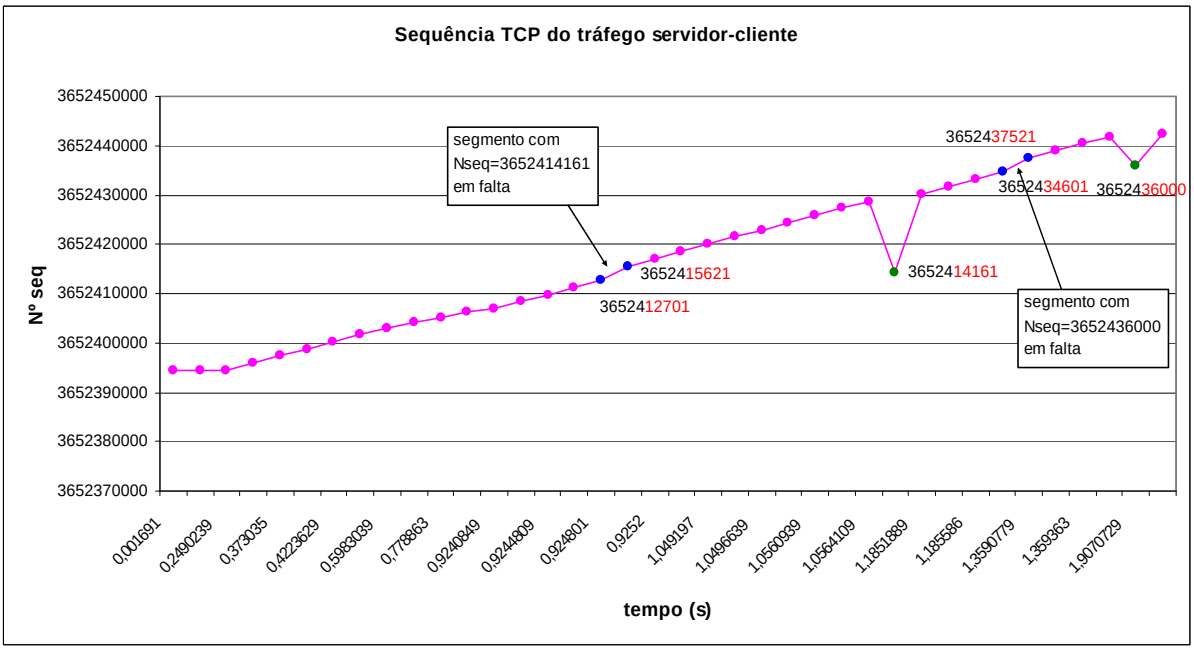


**Figura 14 Sequência TCP do tráfego servidor-cliente**

Com base no excerto do traço e no gráfico da Figura 14 conclui-se que houveram duas retransmissões de segmentos perdidos (descartados), com os nº de seq. 3652414161 e 3652436000. A retransmissão foi feita recorrendo ao mecanismo de retransmissão rápida (para evitar timeout e consequente slow-start). O receptor implementa o mecanismo de controlo de fluxo *selective-reject ARQ*, pelo que quando recebe o segmento em falta aproveita os segmentos consequentes já recebidos.

É possível verificar que o segmento foi perdido com base no nº de seq. e tamanho do segmento imediatamente anterior. A título de exemplo, no caso do primeiro segmento perdido, foi observado o segmento com nº de seq. 3652412701 e tamanho 1460 bytes (= 1500 – 20 cab.IP – 20 cab.TCP), o segmento seguinte esperado era um com seq. 3652414161 (3652412701 + 1460), mas o verificado foi um com seq. 3652415621.

Pelas flags TCP dos dos primeiros segmentos é confirmar que se tratam dos segmentos de estabelecimento da conexão TCP: 2→SYN, 12→ACK+SYN, 10→ACK.

De igual forma é possível confirmar o término da conexão iniciada pelo servidor: 11→ACK+FIN, 10→ACK.

Embora não possa ser dado como regra, verifica-se que os segmentos com dados tem as flags ACK+PSH activas (18). A flag PSH não tem necessariamente, de estar activa, o objectivo é fazer com que os dados sejam entregues à aplicação mesmo antes que o buffer de recepção desta última esteja completamente preenchido, optimizando as aplicações.

## *1.8 Resposta 8*

*Represente a função distribuição complementar do tempo entre chegada de pacotes consecutivos na interface 1. O eixo dos YY deve ser representado em escala logaritmica, de base 10.*

*Represente sobre a mesma curva a função distribuição complementar da distribuição exponencial. O que observa? Comente.*

Nas figuras que se seguem são apresentados dois gráficos, que diferem apenas na apresentação, onde o 2º apresenta a escala das oordenadas segundo uma escala logarítmica de base 10. Em cada um são apresentadas duas curvas: a função distibuição complementar conseguida através do traço e a função distribuição complementar da distribuição exponencial definida por $e^{-\lambda x}$, sendo λ a taxa de chegada de pacotes.
Para estimar λ recorremos ao inverso da média amostral do tempo entre chegadas (E(x)), sendo este último igual a 0,00047585 segundos, assim, vem λ igual a 2101,459 pacotes recebidos/segundo.

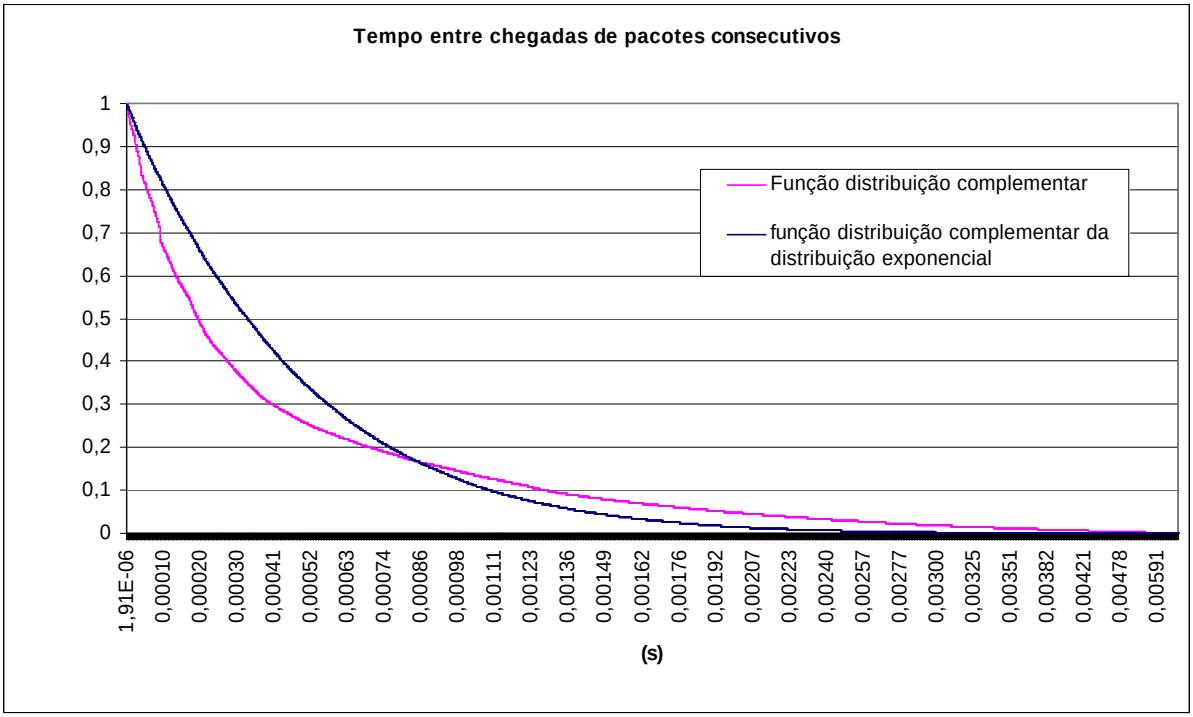


**Figura 15 Tempo de chagada entre pacotes consecutivos**

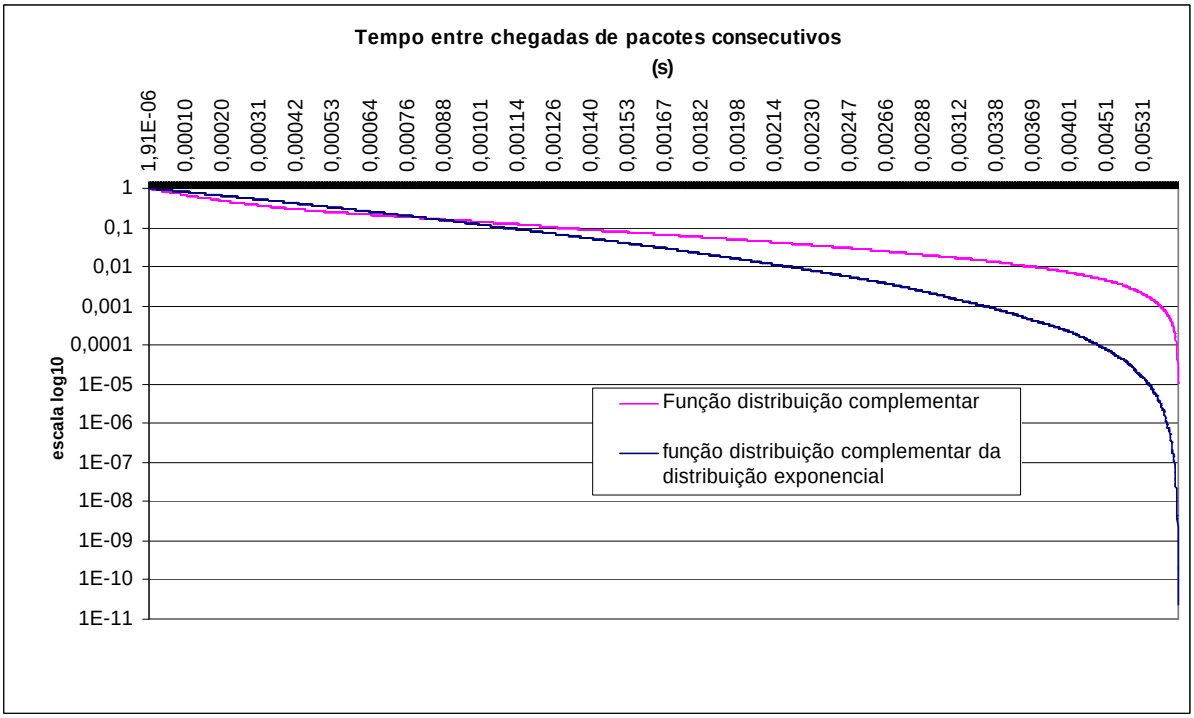


**Figura 16 Tempo de chagada entre pacotes consecutivos (escala logarítmica)**

Ao analisar as duas curvas podemos observar uma grande semelhança, permitindo-nos concluir que o tempo de chegada entre pacotes pode ser modelizado segundo uma distribuição exponencial. Assim sendo, o processo de geração de chegadas de pacotes pode ser modelizado por uma distribuição de Poisson.

## 2 ANEXO – CÓDIGO FONTE DE AMSR_TRAB3.PL

```
#!usr/bin/perl

use strict;

die "Usage: $0 logfile\n" unless @ARGV == 1;

my $logfile = shift;

open(FILE,"<$logfile") or die "Can't open log file.\n";

my @list;

my $count_pacotes; #contador do número total de pacotes no traço
my $count_bytes; #contador do número total de bytes no traço
my %protocolo_num_pacotes; #hash table que contem o número de pacotes que dizem respeito a cada protocolo
my %protocolo_num_bytes; #hash table que contem o número de bytes que dizem respeito a cada protocolo

my %funcao_distribuicao; #hash table que contem o numero de pacotes enviados por numero de bytes
my %funcao_cumulativa; #hash table que o contem, para um dado numero de bytes, o numero de pacotes enviados de comprimento inferior a esse numero de bytes
my %funcao_cumulativa_percentagem; #idem, mas em percentagem do numero total de pacotes enviados
my $total_cumulativo;

my %servico_num_pacotes; #hash table que contem o numero de pacotes enviados que dizem respeito a cada serviço
my %servico_num_pacotes_percentagem; #idem, mas em percentagem do número total de pacotes enviados
my $count_pacotes_sem_porta_bem_conhecida;
my $count_acks_puros;

my $fluxo;
my %fluxos; #possíveis fluxos
my %fluxos_reais; #fluxos - com pelo menos 3 pacotes
my $count_fluxos;
my $count_fluxos_bytes; #total de bytes que correspondem a pacotes pertencentes a fluxos
my $count_fluxos_udp;
my $count_fluxos_tcp;

my %fluxos_bytes;
my $fluxo_cumulativo;
my %fluxos_cumulativo_percentagem;
my $count;

my $tempo_inteiro;
my %debito;
my %debito_100ms;
my %debito_10ms;
my $tempo_actual_100ms=990048350.0;
my $tempo_actual_10ms=990048350.00;

my $fluxo_http;
my $fluxo_http_inverso;
my %todos_fluxos_http;
my %num_sequencia_tempo; #numeros de sequencia de um dado fluxo http que começa no tempo 990048367.932311 em funcao do tempo
my $tempo_relativo;
my $count_http;
my $count_http_inverso;
my $count_http_total;
```

```
my $tempo_entre_chegadas;
my %numero_pacotes_funcao_tempo_chegada;
my $tempo_chegada_pacote_anterior = 0;
my $numero_pacotes_funcao_tempo_chegada_acumulado;
my %numero_pacotes_funcao_tempo_chegada_acumulado;
my %funcao_distribuicao_complementar;
my $count_pacotes_int_1;
my $tempos_acumulados;
my $media_amostral;
my $taxa_chegada;

my $ficheiro_log = "trab3.log";

while (<FILE>) {
	@list=split /\s+/;

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 1
	$count_pacotes++;
	$count_bytes=$count_bytes+$list[5];
	$protocolo_num_pacotes{"$list[4]"}++;
	$protocolo_num_bytes{"$list[4]"}=$protocolo_num_bytes{"$list[4]"}+$list[5];
#-------------------------------------------------------------------------------
-----------

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 2
	$funcao_distribuicao{"$list[5]"}++;
#-------------------------------------------------------------------------------
-----------

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 3
	if (($list[4]==6) || ($list[4]==17)) {
		if ($list[6]==$list[7]) {
			$servico_num_pacotes{"$list[6]"}++;
		}
		else {
			$servico_num_pacotes{"$list[6]"}++;
			$servico_num_pacotes{"$list[7]"}++;
		}
	}

	if (($list[4]==6) || ($list[4]==17)) {
		if (($list[6]>1023) && ($list[7]>1023)) {
			$count_pacotes_sem_porta_bem_conhecida++;
		}
	}

	if (($list[8]==10) && ($list[5]==40) && ($list[4]==6)) {
		$count_acks_puros++;
	}
#-------------------------------------------------------------------------------
-----------

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 4
	$fluxo = "$list[2] $list[3] $list[4] $list[6] $list[7]";

	$fluxos{$fluxo}++;

	if ($fluxos{$fluxo}==3) {
		$fluxos_reais{$fluxo}=$fluxos{$fluxo};
		$count_fluxos++;
```

```
              if ($list[4]==17) {
                          $count_fluxos_udp++;
              }
              if ($list[4]==6) {
                     $count_fluxos_tcp++;
              }
       }
#-------------------------------------------------------------------------------
-----------

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 5
       $fluxos_bytes{$fluxo}=$fluxos_bytes{$fluxo}+$list[5];
#-------------------------------------------------------------------------------
-----------

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 6
       $tempo_inteiro=int($list[1]);
       $debito{"$tempo_inteiro"}=$debito{"$tempo_inteiro"}+$list[5];

       if ($list[1] lt ($tempo_actual_100ms+0.1)) {

       $debito_100ms{"$tempo_actual_100ms"}=$debito_100ms{"$tempo_actual_100ms"}+$l
ist[5];
       }
       else {
              $tempo_actual_100ms += 0.1;
              $debito_100ms{"$tempo_actual_100ms"}=$list[5];
       }
       if ($list[1] lt ($tempo_actual_10ms+0.01)) {

       $debito_10ms{"$tempo_actual_10ms"}=$debito_10ms{"$tempo_actual_10ms"}+$list[
5];
       }
       else {
              $tempo_actual_10ms += 0.01;
              $debito_10ms{"$tempo_actual_10ms"}=$list[5];
       }


#-------------------------------------------------------------------------------
-----------

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 7

       if ($list[1]==990048367.932311) {
              $fluxo_http = "$list[2] $list[3] $list[4] $list[6] $list[7]";
              $fluxo_http_inverso = "$list[3] $list[2] $list[4] $list[7] $list[6]";
       }

       $tempo_relativo = $list[1] - 990048367.932311;
       if (($fluxo eq $fluxo_http_inverso) || ($fluxo eq $fluxo_http)) {
              $count_http_total++;
              $todos_fluxos_http{"$count_http_total"}=$fluxo;
              if ($fluxo eq $fluxo_http_inverso) {
                     $num_sequencia_tempo{"$tempo_relativo"}=$list[9];
                     $count_http_inverso++;
              }
              if ($fluxo eq $fluxo_http) {
                     $count_http++;
              }
       }
```

```
#-------------------------------------------------------------------------------
-----------

#-------------------------------------------------------------------------------
-----------
#PERGUNTA 8
	if ($list[0] == 1) {
		if ($tempo_chegada_pacote_anterior!=0) {
			$tempo_entre_chegadas		=		$list[1]		-
$tempo_chegada_pacote_anterior;

	$numero_pacotes_funcao_tempo_chegada{"$tempo_entre_chegadas"}++;
			$count_pacotes_int_1++;

		}
		$tempo_chegada_pacote_anterior = $list[1];
	}
#-------------------------------------------------------------------------------
-----------
}

open(OUTPUT_FILE, ">$ficheiro_log");

#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 1
#-------------------------------------------------------------------------------
-----------
print OUTPUT_FILE "\nPERGUNTA 1\n";
print	OUTPUT_FILE	"Listagem	de	pacotes	(protocolo)	por	numero	de	pacotes
capturados\n";
#NUMERO DE PACOTES
print OUTPUT_FILE "NUMERO DE PACOTES\n";
do {
	print OUTPUT_FILE "$_ , $protocolo_num_pacotes{$_}\n";
}
foreach	(	sort	{$protocolo_num_pacotes{$b}	<=>	$protocolo_num_pacotes{$a}}	keys
%protocolo_num_pacotes);
#PERCENTAGEM DE PACOTES
print OUTPUT_FILE "PERCENTAGEM DE PACOTES\n";
do {
	$protocolo_num_pacotes{$_}=($protocolo_num_pacotes{$_}/$count_pacotes)*100;
	print OUTPUT_FILE "$_ , $protocolo_num_pacotes{$_} %\n";
}
foreach	(	sort	{$protocolo_num_pacotes{$b}	<=>	$protocolo_num_pacotes{$a}}	keys
%protocolo_num_pacotes);

print OUTPUT_FILE "\nListagem de pacotes (protocolo) por numero de bytes\n";
#NUMERO DE BYTES
print OUTPUT_FILE "NUMERO DE BYTES\n";
do {
	print OUTPUT_FILE "$_ , numero de bytes = $protocolo_num_bytes{$_}\n";
}
foreach	(	sort	{$protocolo_num_bytes{$b}	<=>	$protocolo_num_bytes{$a}}	keys
%protocolo_num_bytes);
#PERCENTAGEM
print OUTPUT_FILE "PERCENTAGEM\n";
do {
	$protocolo_num_bytes{$_}=($protocolo_num_bytes{$_}/$count_bytes)*100;
	print OUTPUT_FILE "$_ , percentagem = $protocolo_num_bytes{$_} %\n";
}
foreach	(	sort	{$protocolo_num_bytes{$b}	<=>	$protocolo_num_bytes{$a}}	keys
%protocolo_num_bytes);

#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 2
#-------------------------------------------------------------------------
```

```
print OUTPUT_FILE "\nPERGUNTA 2\n";
print OUTPUT_FILE "\Funcao de distribuicao\n";
do {
	print OUTPUT_FILE "$_ , $funcao_distribuicao{$_}\n";
}
foreach ( sort {$a <=> $b} keys %funcao_distribuicao);
print OUTPUT_FILE "\nFuncao cumulativa\n";
do {
	$total_cumulativo=$total_cumulativo+$funcao_distribuicao{$_};
	$funcao_cumulativa{$_}=$total_cumulativo;
	$funcao_cumulativa_percentagem{$_}=$funcao_cumulativa{$_}/$count_pacotes*100
;
	print OUTPUT_FILE "$_ , $funcao_cumulativa_percentagem{$_} %\n";
}
foreach ( sort {$a <=> $b} keys %funcao_distribuicao);

#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 3
#-------------------------------------------------------------------------
print OUTPUT_FILE "\nPERGUNTA 3\n";
print  OUTPUT_FILE  "Listagem  de  pacotes  (servico)  por  numero  de  pacotes
capturados\n";
do {
	if (($_== 80) || ($_== 25) || ($_== 21) || ($_== 123) || ($_== 53) || ($_==
20) || ($_== 119)) {
		$servico_num_pacotes_percentagem{$_}                                =
($servico_num_pacotes{$_}/$count_pacotes)*100;
		print    OUTPUT_FILE    "$_    ,    $servico_num_pacotes{$_}    ->
$servico_num_pacotes_percentagem{$_} %\n";
	}
}
foreach  (  sort  {$servico_num_pacotes{$b}  <=>  $servico_num_pacotes{$a}}  keys
%servico_num_pacotes);
print OUTPUT_FILE "O numero de pacotes que nao usam uma porta bem conhecida e,
$count_pacotes_sem_porta_bem_conhecida\n";
print OUTPUT_FILE "O numero de ACKS puros e, $count_acks_puros\n";

#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 4
#-------------------------------------------------------------------------
print OUTPUT_FILE "\nPERGUNTA 4\n";
print OUTPUT_FILE "O numero de fluxos e, $count_fluxos\n";
print OUTPUT_FILE "O numero de fluxos de pacotes TCP e, $count_fluxos_tcp\n";
print OUTPUT_FILE "O numero de fluxos de pacotes UDP e, $count_fluxos_udp\n";

#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 5
#-------------------------------------------------------------------------
print OUTPUT_FILE "\nPERGUNTA 5\n";
do {
	if ($fluxos{$_}>=3) {
		$count_fluxos_bytes = $count_fluxos_bytes + $fluxos_bytes{$_};
	}
}
foreach keys %fluxos;

do {
	$count++;
	$fluxo_cumulativo=$fluxo_cumulativo+$fluxos_bytes{$_};
	$fluxos_cumulativo_percentagem{"$count"}=$fluxo_cumulativo/$count_fluxos_byt
es*100;
	print OUTPUT_FILE "$count , $fluxos_cumulativo_percentagem{$count}\n";
}
foreach ( sort { $fluxos_bytes{$b} <=> $fluxos_bytes{$a} } keys %fluxos_reais);
```

```
#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 6
#-------------------------------------------------------------------------
print OUTPUT_FILE "\nPERGUNTA 6\n";
print OUTPUT_FILE "Debito em funcao do tempo (intervalo de amostragem - 1s)\n";
do {
      print OUTPUT_FILE "$_ , $debito{$_}\n";
}
foreach ( sort { $a <=> $b } keys %debito);
print OUTPUT_FILE "Debito em funcao do tempo (intervalo de amostragem - 100ms)\n";
do {
      $debito_100ms{$_}=$debito_100ms{$_}/0.1;
      printf OUTPUT_FILE "%.1f , $debito_100ms{$_}\n" , $_;
}
foreach ( sort { $a <=> $b } keys %debito_100ms);
print OUTPUT_FILE "Debito em funcao do tempo (intervalo de amostragem - 10ms)\n";
do {
      $debito_10ms{$_}=$debito_10ms{$_}/0.01;
      printf OUTPUT_FILE "%.2f , $debito_10ms{$_}\n" , $_;
}
foreach ( sort { $a <=> $b } keys %debito_10ms);

#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 7
#-------------------------------------------------------------------------------
-----------
print OUTPUT_FILE "\nPERGUNTA 7\n";
print OUTPUT_FILE "\Todos os fluxos da ligacao HTPP!\n";
do {
      print OUTPUT_FILE "$_ , $todos_fluxos_http{$_}\n";
}
foreach ( sort { $a <=> $b } keys %todos_fluxos_http );
print OUTPUT_FILE "cliente -> servidor , $count_http\n";
print OUTPUT_FILE "servidor -> cliente , $count_http_inverso\n";
print OUTPUT_FILE "Numeros de sequencias dos fluxos do servidor para o cliente em
funcao do tempo\n";
do {
      print OUTPUT_FILE "$_ , $num_sequencia_tempo{$_}\n";
}
foreach ( sort { $a <=> $b }  keys %num_sequencia_tempo );

#-------------------------------------------------------------------------------
-----------
#RESULTADO PERGUNTA 8
#-------------------------------------------------------------------------------
-----------
print OUTPUT_FILE "\nPERGUNTA 8\n";
do {
      $numero_pacotes_funcao_tempo_chegada_acumulado=$numero_pacotes_funcao_tempo_
chegada_acumulado+$numero_pacotes_funcao_tempo_chegada{$_};
      $numero_pacotes_funcao_tempo_chegada_acumulado{$_}=$numero_pacotes_funcao_te
mpo_chegada_acumulado;
      $funcao_distribuicao_complementar{$_}=1-
($numero_pacotes_funcao_tempo_chegada_acumulado{$_}/($count_pacotes_int_1));
      print OUTPUT_FILE "$_ , $funcao_distribuicao_complementar{$_}\n";
      $tempos_acumulados=$_*$numero_pacotes_funcao_tempo_chegada{$_}+$tempos_acumu
lados;
}
foreach ( sort { $a <=> $b }  keys %numero_pacotes_funcao_tempo_chegada );

$media_amostral=$tempos_acumulados/$count_pacotes_int_1;
$taxa_chegada=1/$media_amostral;
print OUTPUT_FILE "media amostral , $media_amostral\n";
print OUTPUT_FILE "taxa chegada , $taxa_chegada";
```