互联网数据流特征分析

概率与统计 实验报告

Probability and Statistics

Experiment Report

指导教师：　范正平

学科专业：　软件工程

班级：　　教务四班

小组成员：　熊永琦　16340258

徐伟元　16340261

姚俊威　16340276

熊秭鉴　16340259

所属学院：数据科学与计算机学院

一、 数据预分析

我们将数据包压缩文件解压，并将其解压过后的文件导入到 wireshark。在分析的过程中，我们首先利用 wireshark 自带的统计分析工具对数据进行预分析，在确定一些数据的基本特点后，我们利用 wireshark 的命令行工具 tshark 解析 pcap 数据包，生成文本文件，以方便地导入 Matlab，这为我们下一步的工作提供了帮助。

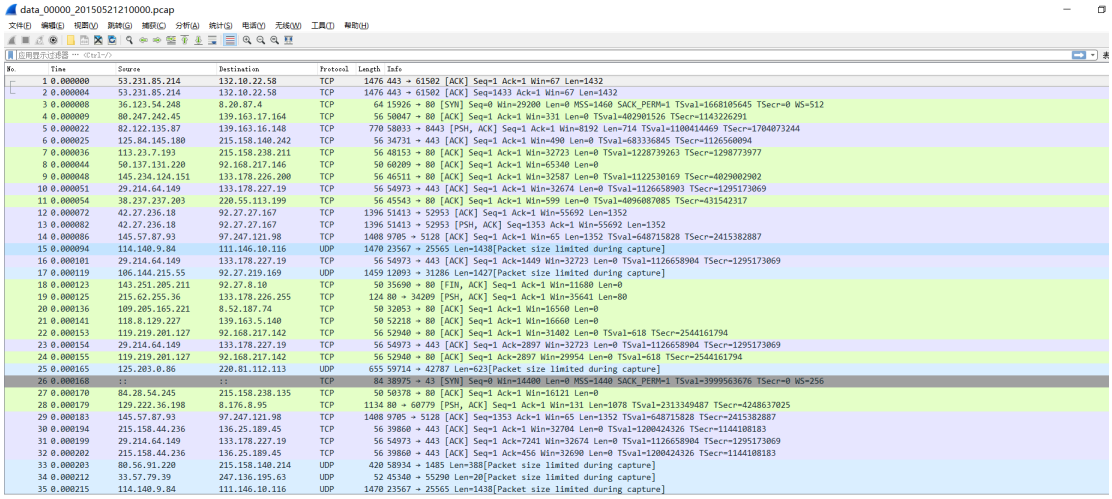

图 1-1 利用 Wireshark 进行预分析（数据包列表）

协议	按分组百分比	分组	按字节百分比	字节	比特/秒	End Packets	End Bytes	End Bits/s
Frame	100.0	17721707	59.2	837796988	111 M	0	0	0
Raw packet data	100.0	17721707	59.2	837796988	111 M	0	0	0
Internet Protocol Version 6	0.1	15318	0.0	612720	81 k	24	960	128
User Datagram Protocol	0.0	2158	0.0	17264	2301	0	0	0
Short Frame	0.0	2158	0.0	0	0	2158	0	0
Transmission Control Protocol	0.1	10500	0.0	230144	30 k	10480	229344	30 k
Short Frame	0.0	20	0.0	0	0	20	0	0
Internet Control Message Protocol v6	0.0	2618	0.0	20944	2792	2577	20616	2748
Short Frame	0.0	41	0.0	0	0	41	0	0
Data	0.0	18	0.0	144	19	18	144	19
Internet Protocol Version 4	99.9	17706389	25.0	354127780	47 M	3459	69180	9224
User Datagram Protocol	10.5	1854083	1.0	14832664	1977 k	113	904	120
Vuze DHT Protocol	0.0	356	0.0	0	0	0	0	0
Virtual eXtensible Local Area Network	0.0	1	0.0	0	0	0	0	0
UTRAN IuBC interface SABP signaling	0.0	8	0.0	0	0	0	0	0
UDP Encapsulation of IPsec Packets	0.1	16840	0.0	0	0	0	0	0
Teredo IPv6 over UDP tunneling	0.0	258	0.0	0	0	0	0	0
Short Frame	10.5	1853970	0.0	0	0	1853970	0	0
SEBEK - Kernel Data Capture	0.0	13	0.0	0	0	0	0	0
Remote Procedure Call	0.0	2	0.0	0	0	0	0	0
Real Data Transport	0.0	5	0.0	0	0	0	0	0
QuakeWorld Network Protocol	0.0	3	0.0	0	0	0	0	0
Quake III Arena Network Protocol	0.0	75	0.0	0	0	0	0	0
PacketCable	0.0	8	0.0	0	0	0	0	0
OpenVPN Protocol	0.1	23854	0.0	0	0	0	0	0
OMA UserPlane Location Protocol	0.0	7	0.0	0	0	0	0	0
Media-Independent Handover	0.0	1	0.0	0	0	0	0	0
LWAPP Layer 3 Packet	0.0	3	0.0	0	0	0	0	0
Locator/ID Separation Protocol (Data)	0.0	1	0.0	0	0	0	0	0
Kontiki Delivery Protocol	0.0	1	0.0	0	0	0	0	0
kNet Protocol	0.0	4	0.0	0	0	0	0	0
IP Virtual Services Sync Daemon	0.0	1	0.0	0	0	0	0	0
H323-MESSAGES	0.0	5	0.0	0	0	0	0	0
EtherNet/IP (Industrial Protocol)	0.0	1	0.0	0	0	0	0	0
Endpoint Handlespace Redundancy Protocol	0.0	6	0.0	0	0	0	0	0
Echo	0.0	6	0.0	0	0	0	0	0
Dropbox LAN sync Discovery Protocol	0.0	2	0.0	0	0	0	0	0
DOF Protocol Stack	0.0	5	0.0	0	0	0	0	0
Distributed Interactive Simulation	0.0	70	0.0	0	0	0	0	0
DirectPlay 8 protocol	0.0	3	0.0	0	0	0	0	0
Daytime Protocol	0.0	2	0.0	0	0	0	0	0
Datagram Transport Layer Security	0.0	6	0.0	0	0	0	0	0
Control And Provisioning of Wireless Access Points - Control	0.0	1	0.0	0	0	0	0	0
Character Generator Protocol	0.0	4	0.0	0	0	0	0	0
Blubster/Piolet MANOLITO Protocol	0.0	1	0.0	0	0	0	0	0
AX/4000 Test Block	0.0	1	0.0	0	0	0	0	0
Aruba Discovery Protocol	0.0	1999	0.0	0	0	0	0	0
Transmission Control Protocol	89.2	15802772	33.0	467643076	62 M	15802772	467643076	62 M

图 1-2 利用 Wireshark 进行预分析（数据分级）

从协议分级中我们可以明显的看出，**TCP 是互联网数据包的绝对主力**，占据了 89.2%的 ipv4 数据包总量，而另一大头则是 UDP，却仅有 10.5%的包量，因此我们主要针对 TCP 包进行分析。同时，从数据包列表界面我们还能了解到：在所有的数据包中，**有相当一部分**

包是仅有报头的空包，这些包由 TCP 的三次握手等协议规定的数据空包动作产生，而这部分包的数量，以及他们反映的问题在预处理部分有详细的解析。

二、 数据预处理

为了提高数据的代表性，减少分析数据的时间，我们对数据进行了预处理。预处理分两步进行：首先，我们选择**协议为 TCP 且不为空的包**，选择 TCP 协议，是因为其是当前互联网最主流且有绝对优势的协议，分析他很大程度上可以**反映互联网整体状态**，并且可以**排除 UDP 存在时，两种协议间数据的互相干扰**。然后，我们使用 Matlab 中基于**红黑树**实现的 map 容器，高效地对所有数据包进行了分组，分组的依据是起始 IP 与目标 IP。并筛除其中包量不足 100 的所有组分，最终得到了 7300 余组，共计 800 余万条的信息，并将其储存在 midProduct.txt 文件中。

在这个过程中，我们还得到了一些阶段性的结论：

在第一步，也就是协议和包大小过滤中，我们发现包的总数由原来的 1770 万条锐减到 960 万条，结合之前预分析过程中的结论，我们不难发现，在 1770 万条记录中竟然有 **560 万条记录均为空包**，也就是说，只是 TCP 报头构成的空包，就占了整个数据包集合的 **31.6%**。这反映出，在现代网络数据流中，连接的建立和中断是极为频繁的，这对服务器的 I / O 性能，以及多连接维持的性能优很高的要求——这刚好能解释为什么现代服务器的热门架构更多擅长于**高并发、多连接**的运行环境（如 Nodejs、Apache），因为网络环境的确如此，因而她们成为热门。

而在第二步中，我们过滤了所有包量不足 100 的 IP 对间的数据包，然而数据量仅减少了 100 万，这说明大部分数据包是在重复的 IP 间传输的。据此我们分析，这应该归结于现代网络架构：用户之间的连接大多通过**数据中心**的中转，因而所有用户间的数据包都实际上是发向某些固定的大型服务器的。同时，运营商为了节约成本，大多会将大量用户**集束到同一网络出口**，这也使得 IP 的重复率大大增加。

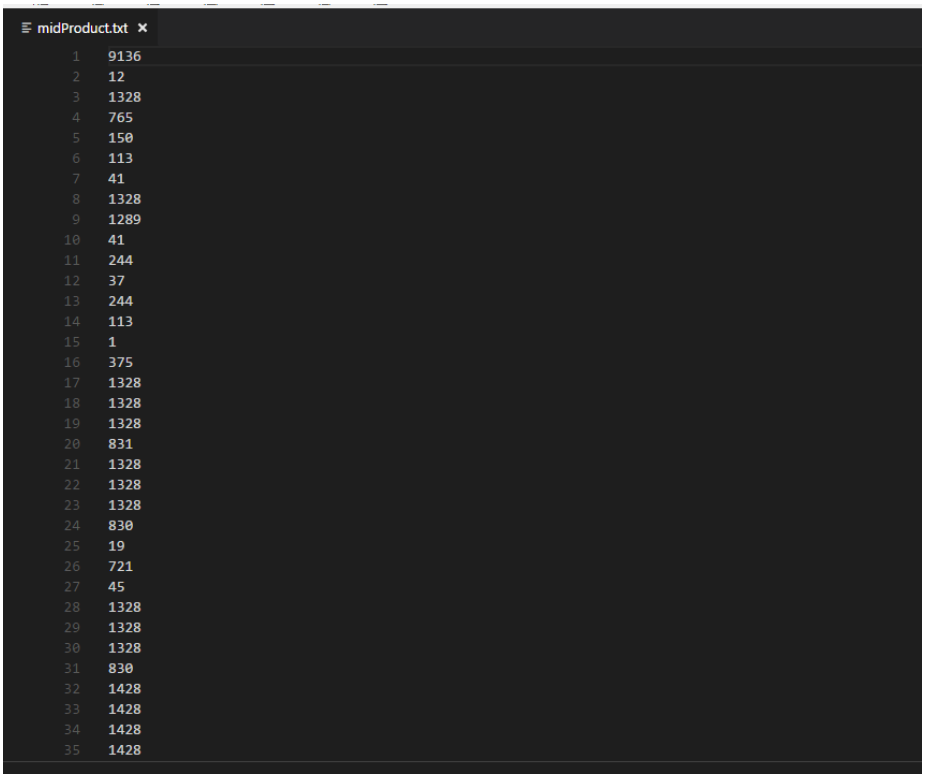

图 2-1 预处理后输出的文件

三、 分析过程

1） 存储数据

为了导入在先前解析和预处理的数据，并在内存中按一定便于分析的格式存储。在matlab 里面，我们构造一个名为 build 的函数。其作用是读取之前生成的预处理数据文件，并按照规定的格式将数据存储在一个二维数组里面，然后将这个数组作为一个传出的变量以供使用。在这个二维数组里面，我们规定数组每一行的第一个数字表示的是一组 IP 地址的数据包个数 N，之后 N 个数字是这组 IP 地址下数据包大小的合集。

针对文件的读取操作，采用的是 importdata 方法：

```
function matrix = build()
delimiterIn   = '';
headerlinesIn = 0;
A = importdata('midProduct.txt', delimiterIn, headerlinesIn);
count = 0;
max = 0;
i = 1;
while(i <= size(A,1))
    count = count + 1;
    if max < A(i)
        max = A(i);
    end
    if i + A(i) + 1 > size(A,1)
        break;
    end
    i = i + A(i) + 1;
end
matrix = zeros(count,max+1);
num = 1;
j = 1;
while(j <= size(A,1))
    matrix(num,1) = A(j);
   temp = 2;
   k = j + 1;
   while(k <= j + A(j))
        matrix(num,temp) = A(k);
        temp = temp + 1;
        k = k + 1;
    end
    num = num + 1;
    j = j + A(j) + 1;
end

end
```

2) 分布拟合检验

对于这个二维数组，我们每次读取一行，然后将数据包作为样本空间，进行各分布（我们在概统课上学过的分布）的分布拟合检验。通过样本的经验分布函数与给定分布函数的比较推断每一个样本是否来自给定的分布函数的总体。容量为 n 的样本的经验分布函数记为 Fn(x)，可由样本中小于 x 的数据所占的比例得到，给定分布函数记为 G(x)，构造的统计量为，即两个分布函数之差的最大值，对于假设 H0：总体服从给定的分布 G(x)，及给定的，根据 Dn 的极限分布确定统计量关于是否接受 H0 的数量界限。在本次试验中，我们将置信水平设置为 95%，其中置信水平尤其高者选择出来用于绘制图形。

编辑器 - E:\Sharing\pcap2matlab-1.03\p_judge.m

pcap2matlab.m | parsepcap.m | preprocess.m | build.m | p_judge.m

```
result.exponential = [0, 0, 0];

for row = 1 : r
    A = data(row, 2 : data(row,1) + 1);

    % Normal Distribution
    [mean, sd]=normfit(A);
    p1=normcdf(A, mean, sd);
    [H1,p] = kstest(A, [A',p1'], alpha);
    if H1==0
        result.distribution(1) = result.distribution(1) + 1;
        result.distribution(2) = result.distribution(2) + 1;
        if p >= 0.97 && result.normal(1) < 3 && row <= 500000 && row >= 1
            result.normal(result.normal(1) + 2) = row;
            result.normal(1) = result.normal(1) + 1;
        end
    end

    % Gamma Distribution
    phat=gamfit(A, alpha)
    p2=gamcdf(A,phat(1),phat(2));
    [H2,p] = kstest(A, [A,p2], alpha);
    if H2==0
        result.distribution(1) = result.distribution(1) + 1;
        result.distribution(3) = result.distribution(3) + 1;

        if p >= 0.97 && result.gamma(1) < 3 && row < 1000000 && row > 500000
            result.gamma(result.gamma(1) + 2) = row;
            result.gamma(1) = result.gamma(1) + 1;
        end
    end
```

图 3-1 分布判定函数实现

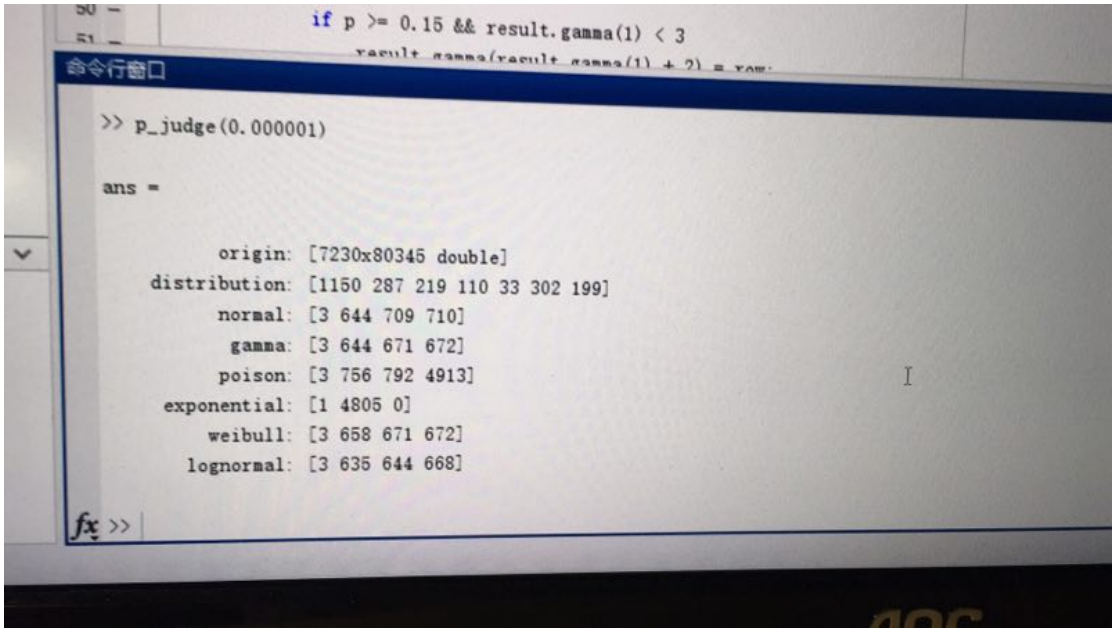

图 3-2 分布判定结果

3）统计分布数据，并导出

在完成数据的处理后，我们利用 Matlab 的图形化工具对其进行了相应的处理。我们将分布拟合检验中得到的拟合参数导入 Matlab 的 pdf（概率密度函数）生成函数，得到对应的离散点集（一个一维矩阵），再借由 plot 模块进行描点画图，在矩阵的尺寸足够大的情况下，这样得到的折线图可近似为曲线图。

四、 分析结果

1、总体概况

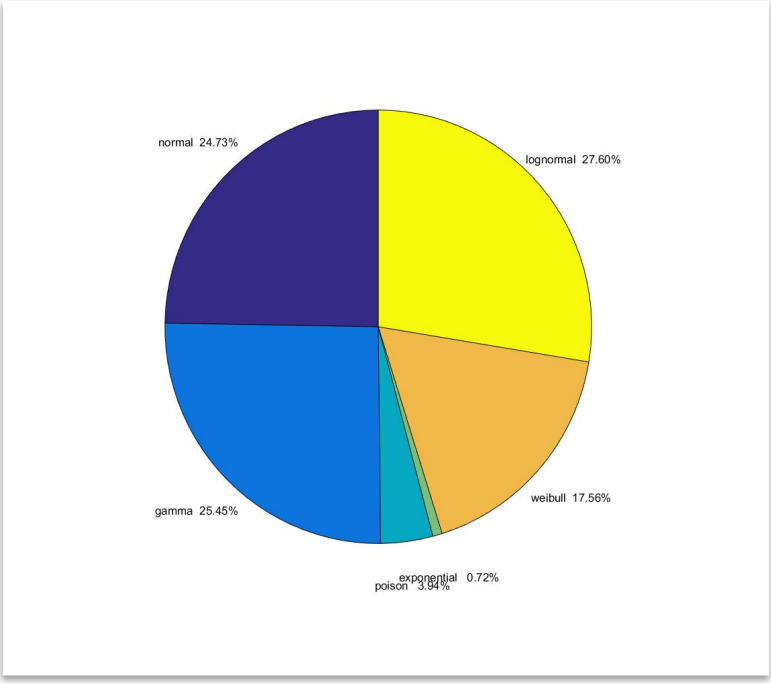

图 4-1 分布占比图

在处理完分析的结果后，我们根据结果绘制了多个图像。首先，是各分布的占比图。从图中我们不难发现，正态、伽马、对数正态和韦布尔分布几乎占据了所有的情况，其中，伽马、正态和对数正态打得平分秋色，占据了超过 3/4 的总量。在下面的板块，我们将对占比较高的几个分布进行解析。

2、主要分布及其成因解析

特别注意：以下所有分布图的横轴均为数据包大小(Length)，纵轴均为由数据包个数的相对频率概率

I） 指数分布与伽马分布

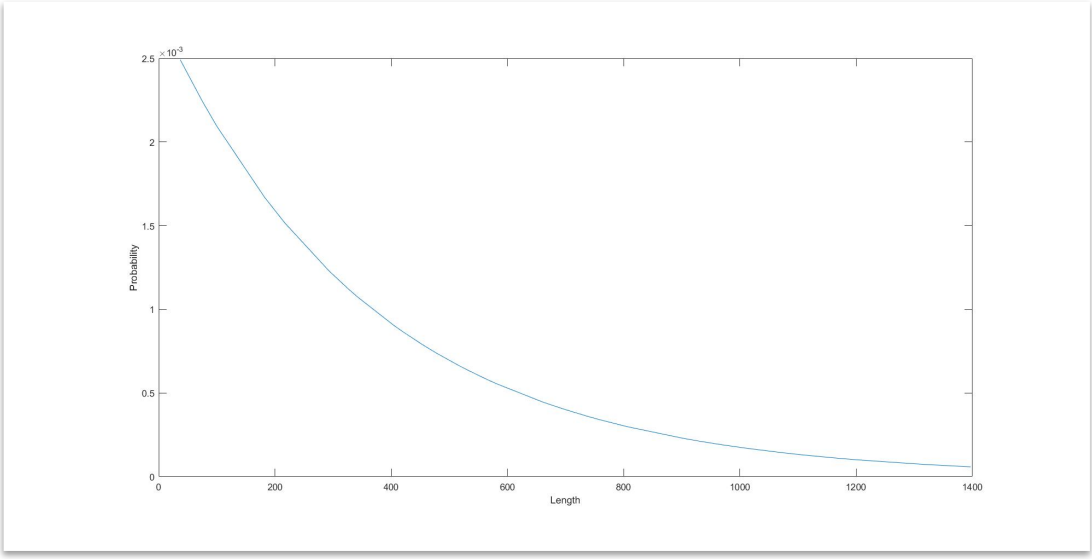

图 4-2 绘制的指数分布图

为什么要把最低的指数分布放在开头呢？因为这次的数据实在是亏待他了。。**在一开始，我们过滤了所有的空包**，这导致很多符合指数分布的数据被洗掉，否则指数分布也应该是一个分布大头。不过他的结论已经在预处理阶段献上了，也算圆满了。

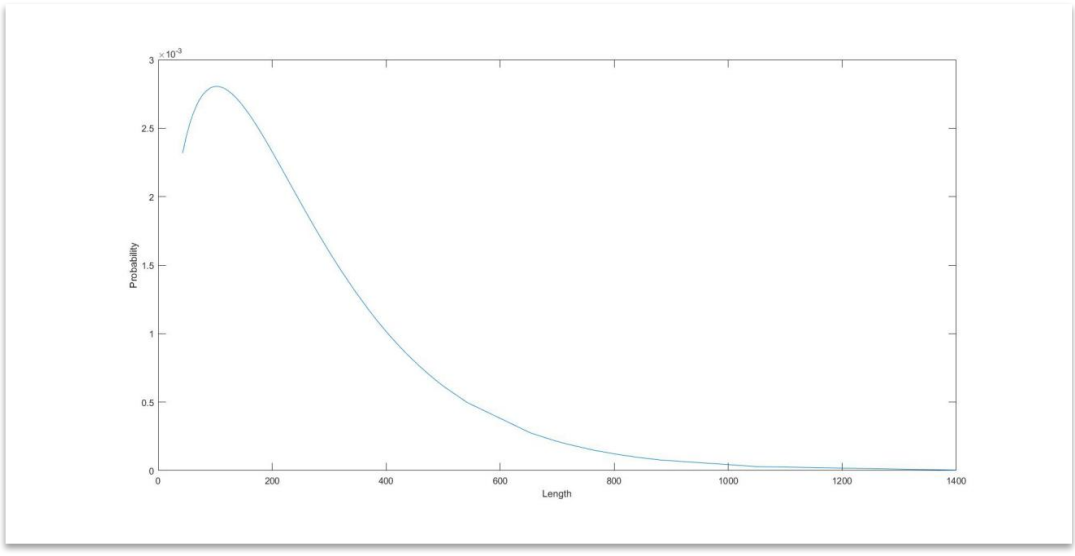

图 4-2 绘制的伽马分布图

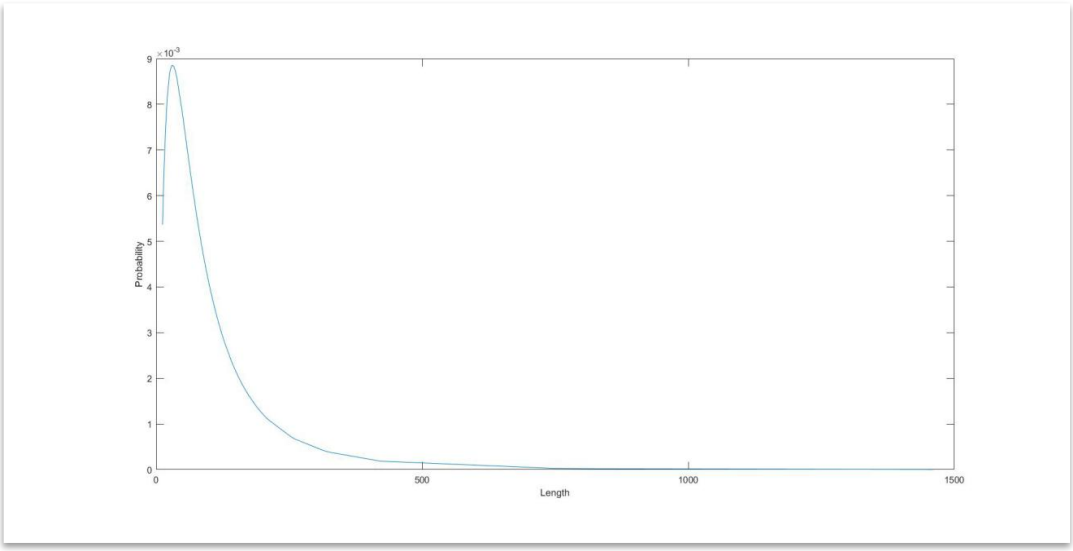

图 4-4 绘制的对数正态分布图

那么为什么要把伽马分布、对数正态分布和指数分布一起说呢？因为它们**其实就是砍掉了空包数据后的指数分布**……明显看出，在左侧数据包量又一个明显的上升后下降，对比指数分布的图像我们容易发现，它正是缺失了空包部分，因而可以预见的是，如果没有洗掉空包数据，那么指数分布很可能成为占比最高的分布。它们很好的反映了**当今高并发、多连接而数据量小的网络环境**，与现代服务器架构理念吻合良好。

II） 正态分布与韦布尔分布

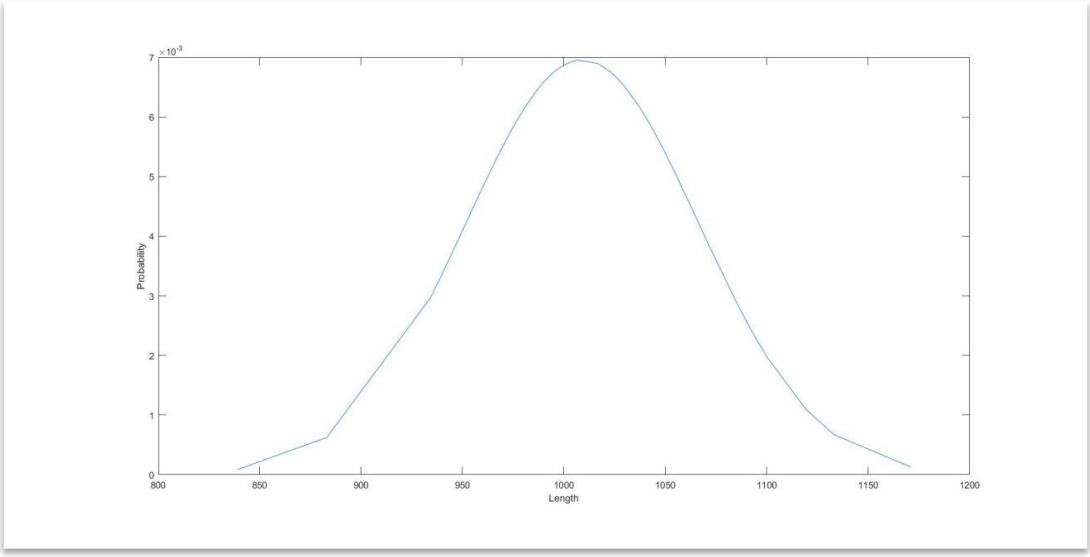

图 4-5 绘制的正态分布图

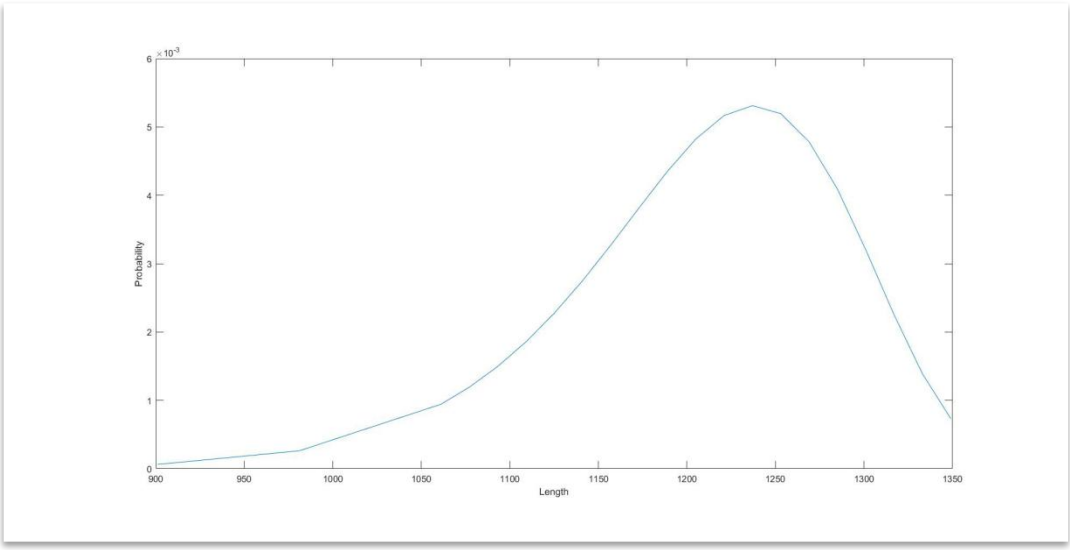

图 4-6 绘制的韦布尔分布图

作为分布的另外另外两个大头，正态分布和韦布尔分布也占了分布占比图的大半边天，它们都反映了互联网数据流的另一个大头：大数据流的传输。

从图像中我们清晰的看到，数据包一开始的占比很小（因为洗掉了空包，这个特征更加明显），然后逐渐上升，在 1k 左右的长度时达到峰值，再下降。在这里，我们要首先了解一个机制：**数据包截段**。

数据包本身是有大小限制的，因而不是所有的数据都能通过一个数据包传输完成，对于那些总量极大的数据，应用层通常会采用数据包截段的方式，将其以多个包的形式发送。这个截断的大小根据应用的不同而有所差异，**但一般集中在 800～1400 的长度**，这和我们的分布图不谋而合。

因而，这两种分布实际上很好地反映了互联网数据流中的另一个大头：大数据量传输，虽然现代网络正在不断向高并发的方向前进，但是大数据传输的需求和压力是始终存在且不可忽视的，在搭建网络系统，分析阻塞模型的过程中，这仍是一个不可忽视的因素。

五、 结语

通过对上述分布模型和实际情况的分析，我们认为，互联网在这份数据采集的时期实际上是处在一个**转型**的阶段。在当前阶段，它的特点是：网络中不仅存在由**大数据流**构成的大量同等大小的大数据包，还有极多的由小数据包，乃至空包组成的**零散数据流**。前者实际上是互联网早期网络流的显著特征，而后者，则是 WEB 技术突飞猛进后带来的。纵览当下的网络巨头：阿里、腾讯、百度，它们很少有大数据量的传输业务，取而代之的是零散数据流的极大膨胀。以淘宝为例，一次购买业务，并不需要太多的数据流量，但是淘宝的核心在于能够承载数十亿这样的零散数据，这也正是互联网向网络服务提供商提出的新的挑战。新生代的网络，将会在原先对**大数据量传输**的要求的基础上，进一步需要服务器能够承载**高并发、多连接数**的网络连接。在阻塞分析方面，将逐步由流量限制导致的阻塞，变为数据包个数限制导致的阻塞，这给中央服务器的转发效率又提出了更高的要求……

总的来说，这次实验让我们体会到了概率论真正的魅力——通过一些早期的数据，我们便能摸清现代互联网发展的脉络，这不得不说是一件令人兴奋的事情。相信我们会带着这份喜悦与动力，向更深的科学领域不断探索。

六、 加分项目

1、多文件处理

在预处理过程中，我们考虑了多文件读取这个问题。很多情况下，我们的数据会太过庞大以至于需要分成数个包进行存储，因此，多文件处理显得尤为必要。在预处理的函数中，我们可以选择传入一组路径，函数将会读取该组中的每一个路径指向的文件，并预处理汇总为一个结果数据交由后续步骤处理。

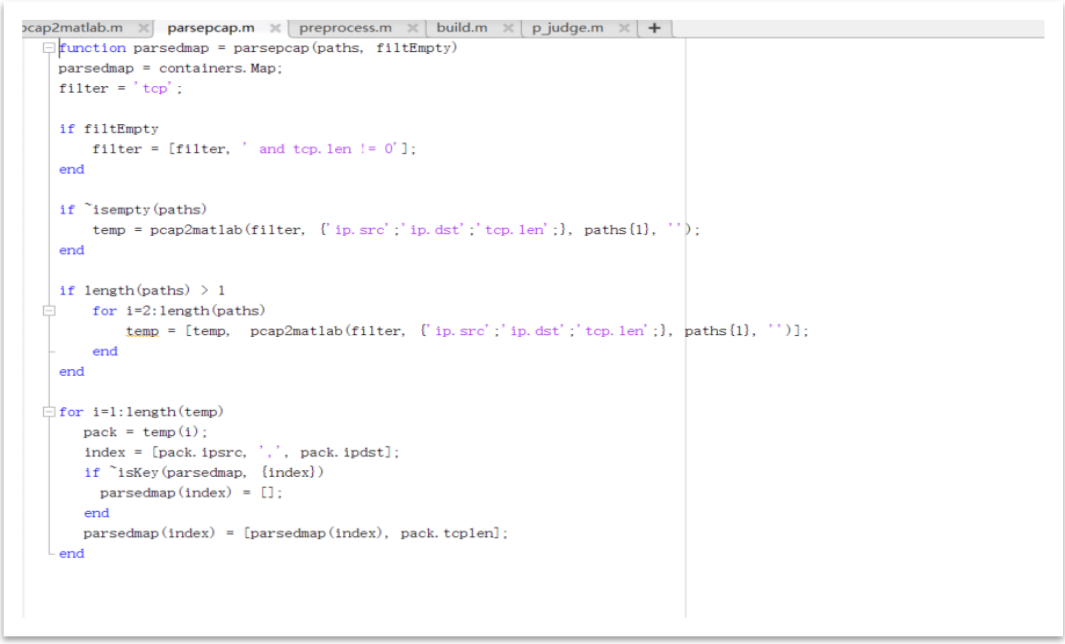

图 6-1 预处理函数中的多文件处理

2、高效率分析

在本次项目中，我们采用了多种方式提高数据的处理效率和效果。首先是**数据预处理**，通过数据预处理，我们可以降低数据的总量，提高数据的针对性，降低无关数据或垃圾数据的干扰，从而更快、更准确的把握数据的特点。其次是**算法优化**，在数据预处理过程中，我们使用 map 数据结构的**红黑树**构建了平衡二叉树，并通过其高效地对我们的数据进行分组。在之后的分析中，我们采用**矩阵**的模式进行数据分析和数据传递，这使得分组信息留存并持续的对后续算法进行优化，使得查找过程的耗时缩短到常量级。