BitTorrent协议概述与实现

陈鸿峥* 冯家苇* 符尧* 傅畅*

数据科学与计算机学院 计算机类

17341015，17341035，17341037，17341038

摘 要 对等网络(P2P)具有去中心化、扩展性高、可靠性高等特点，近年来广泛应用在多媒体、比特币、区块链等领域，成为步入网络3.0时代一项不可或缺的技术。而BitTorrent作为快速的文件共享传输协议，占据了互联网上35%的流量，在对等网络中占据着举足轻重的地位。本文将对BitTorrent协议进行研究，对其概念进行深入剖析与阐述，同时对其进行代码实现，为之后的研究者提供一定的参考。

关键词 对等网络(P2P)，BitTorrent协议

1 引言

随着大数据时代的到来，人们在互联网上交互的信息越来越多，传统的客户端-服务器端(client-server)模型已逐渐体现出弊端—由于中心化的单一服务器端的特点，越来越多的客户端在连入服务器时将存在严重的带宽问题。而对等网络(peer-to-peer, P2P)具有去中心化的特点，在近年来愈发得到人们的关注 [1]。

如图1所示，传统的中心化模型在客户与服务器之间存在十分清晰的界线，客户只能从服务器端下载内容。当存在大量的客户端时，服务器端将面临巨大的负载压力。受限于网络带宽，数据在客户端与与服务器端的传输时间将大幅增加。而去中心化的对等网络中，每一台主机既可以作为客户端，也可以作为服务器端，主机与主机之间不存在任何区别，因而称为对等实体。当从其他对等实体处下载数据时，该对等实体就是客户端；反之，上传数据并为其他对等实体提供资源的即为服务器端。由于数据在对等网络中采用分布式存储，故要获取某些特定数据往往要从多个对等实体处获取，从而实现负载均衡，大大减轻了单一主机的带宽压力，获得更高的数据传输性能。同时，对等网络还具有高可扩展性、高可靠性、高安全性等无可比拟的优势，在多媒体 [2]、比特币 [3]、区块链 [4]等方面得到了广泛应用，已成为步入网络3.0时代一项不可或缺的技术 [5]。

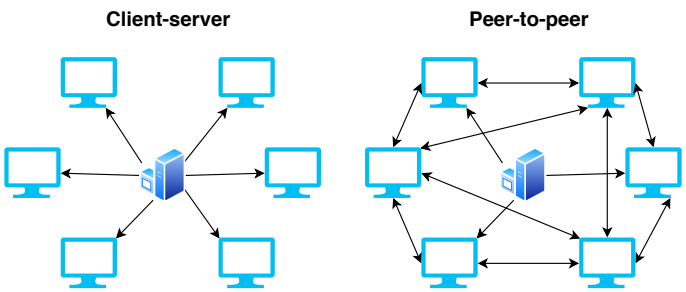

图 1: 客户端-服务器端网络和对等网络示意图

*按学号顺序排序，不代表贡献大小！具体分工如下：陈鸿峥负责论文撰写及文章插图制作；冯家苇负责代码编写与PPT制作；符尧负责代码编写与资料查找；傅畅负责代码编写与资料收集。

而比特洪流(BitTorrent, BT)协议 [6]作为去中心化的快速的文件共享、传输协议，在对等网络中占据十分重要的地位。英国网络分析公司CacheLogic的调查指出，BitTorrent占据了互联网上约35%的流量，并且有数亿用户每天都在使用 [7]。大量的应用都采用了BitTorrent协议，如迅雷 [8]和μTorrent [9]运用BT协议进行文件传输，Facebook [10]和Twitter [11]都采用BT协议进行内容分发，Amazon S3则内置了BT协议进行分布式的文件存储。由此可以看出BitTorrent协议的重要性，故本文将对BT协议进行详细阐述，同时给出其实现方法供后人参考。

2 BitTorrent协议

BitTorrent协议是一个文件分发协议，它使用URL来定位内容并且能够与互联网进行无缝衔接。它相比纯HTTP的好处在于当同个文件的多个下载同时进行时，下载者可以通过上传自己已获得的部分文件供其他人下载，从而实现快速的大文件传输。

BitTorrent协议最早由Bram Cohen于2001年提出，最初只能连接两个实体，传输速度也十分受限 [12]。随着互联网的不断发展，BitTorrent协议也不断更新及完善，现在已能够实现大规模应用，并且充分利用网络带宽，实现快速的文件传输。本文将针对2017年的协议规范(BitTorrent Enhancement Proposals, BEPs) [13]进行阐述[1]，先介绍BitTorrent的基本概念，然后逐一阐述每一个环节的内容，再对完整的文件分享流程进行叙述，最后对具体的算法进行介绍。

2.1 概述

在BitTorrent协议中主要有三方面的内容，首先是以B编码的元信息文件（见第2.2节和第2.3节），其次是参与协议工作的各类实体。

对等实体可以分为播种者(seeder)和吸血者(leecher)。播种者即将自己的文件做成种子，然后对外进行分享。而吸血者仅仅下载别人的文件，而不自己做种，其他人将无法从它这里获得好处。

还有一种实体是追踪者(tracker)，它记录某个文件在哪些播种者中存在副本，并将这些信息告知需要下载文件的对等实体。

关于对等实体的内容见第2.4节，追踪者的内容见第2.5节，BitTorrent协议的具体流程见第2.6节，具体协议内的算法见第2.7节和第2.8节。

2.2 B编码

B编码(Bencoding)是一种数据紧密存储的方式，具体的格式如表1所示。注意任意两项之间没有任何空格，仅仅通过不同的格式区分。列表的内容可以是任意不同类型的组合，即列表内可以嵌套列表也可以包含字符串或字典。

表 1: B编码的格式

类型	格式	例子
字符串	<length>:<data>	7:network
整数	i<integer>e	i3e
列表	l<contents>e	l8:advanced7:networke
字典	d<keys><values>e	d3:onei1e3:twoi2e5:threei3e4:fouri4ee

[1]注意这是BT协议公开的最新版本

2.3 元信息文件

元信息(metainfo)文件，即我们常见的`.torrent`格式的文件或种子文件，用B编码存储，包含了以下信息：

- 公告(announce)：追踪者的url地址，详情见第2.6节。
- 信息(info)：一个字典，其中键值如下，都采用UTF-8编码。
  - 文件名称(name)：一个字符串。如果涉及多个文件或文件夹，则在信息中还包含路径名称(path)。
  - 片段长度(piece length)：每一个片段的字节数。为了传输方便，BitTorrent协议将文件分为等大的片段（除了最后一段）。每一个文件片段的长度通常是2的指数次幂，目前默认为$2^{18} = 256$字节。
  - 片段(piece)：每20个字节代表一个片段的SHA1哈希值，总长度总为20的倍数。
  - 其他可选项：如公告列表、创建日期、创建者、评论等。

如图2所示，元信息文件的格式还是比较清晰的。红色框标注的部分为键的名称，后面紧跟的为键值。

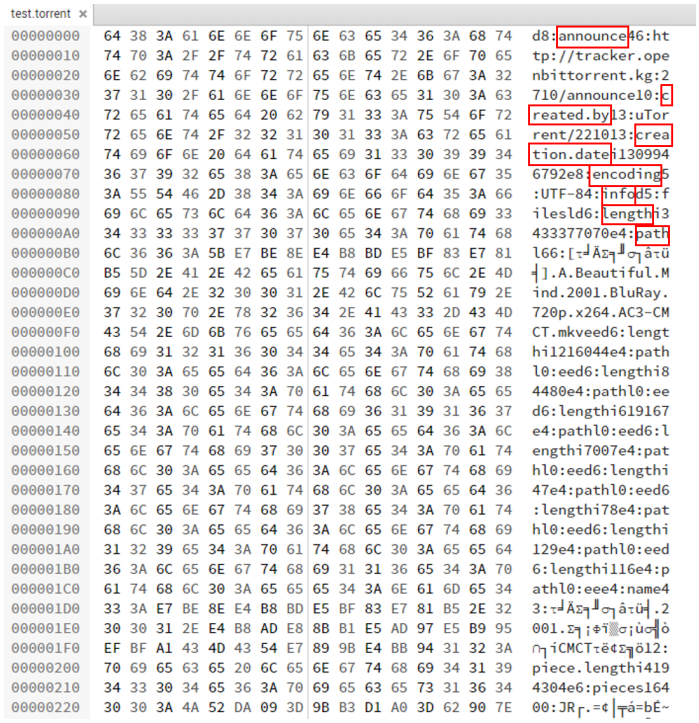

图 2: Torrent测试文件

2.4 对等实体互连协议

对等实体互连协议(peer wire protocol)构建在TCP或μTP协议 [14]上，其连接都是全双工的。

当一个对等实体下载完一个片段并查验其哈希值匹配后，它将对所有实体公告它已经拥有该片段，从而其他实体可以从它这里下载该片段。

对等实体连接两端都包含以下两种不同的状态，用两个二进制位存储：

- 阻塞(choked)：若某一客户端被远端的播种者阻塞，则该客户端无法从该播种者处获取片段。
- 感兴趣(interested)：若某一客户端对远端的播种者的片段感兴趣，则该客户端将在未阻塞时从该播种者处获取片段。

数据传输仅仅发生在一方感兴趣，另一方没有阻塞时。

对等实体之间建立连接需要进行握手确认，其握手信号如下所示。

```
<pstrlen><pstr><reserved><info_hash><peer_id>
```

各字段含义如表2所示。

表 2: 对等实体间握手信号字段含义

字段名	描述
pstrlen	pstr的长度，目前设为19
pstr	协议标识符字符串 目前用"BitTorrent protocol"作为标识
reserved	8个保留字节
info_hash	元信息文件中信息键值的20字节SHA1哈希编码
peer_id	每个实体独立的ID编号，20字节字符串

2.5 追踪者传输协议

追踪者一般采用HTTP协议，分为请求和回复两个阶段。

请求阶段，对等实体通过HTTP GET指令向追踪者发送请求，其中GET的参数如表3所示。

表 3: 追踪者HTTP协议GET请求

参数名	描述
info_hash	元信息文件中的20字节SHA1哈希值
peer_id	20字节客户端ID字符串
port	客户端监听的端口号，默认保留端口为6881-6889
uploaded	当前已上传总量，以十进制ASCII编码
downloaded	当前已下载总量，以十进制ASCII编码
numwant	请求对等实体数目，默认为50

回复阶段，追踪者通过HTTP协议向对等实体回复信息，具体参数如表4所示。

表 4: 追踪者HTTP协议回复

参数名	描述
complete	播种者的数目
incomplete	非播种者的数目
peers	一个字典，包括对等实体ID、IP地址及端口号

与追踪者的握手信号如表5所示。其中Bitfield同样用来指代当前已经被成功下载的片段编号，需要在握手信号已经完成，还未进行其他消息发送之前进行告知。 X代表位域长，位域的最高位代表片段0。而Have则在建立连接后用来指代当前已经被成功下载的片段编号，同样需要发送给追踪者，方便获取新的播种者名单。

表 5: 追踪者握手信号含义

握手信号	<length prefix>	<message ID>	<payload>
Keep-alive	0000	0	无
Choke	0001	0	无
Unchoke	0001	1	无
Interested	0001	2	无
Not-interested	0001	3	无
Have	0005	4	片段编号
Bitfield	0001+X	5	位域
Request	0013	6	<index><begin><length>
Piece	0009+X	7	<index><begin><block>
Cancel	0013	8	<index><begin><length>
port	0003	9	<listen-port>

由于HTTP协议基于TCP协议，需要连接始终保持开启，同时导致大量数据开销，故为了减少开销，BitTorrent的追踪者协议也可以基于UDP协议进行，这在BEP15 [15]中进行了规定。

另一方面，由于单一的追踪者已无法满足这么多对等实体的需求，故现在更多采用分布式稀疏哈希表(Distributed sloppy Hash Table, DHT)用来存储对等实体的通信信息。每个对等实体都将成为一个追踪者，这在BEP5 [16]中进行了规定。

2.6 文件共享机理

BitTorrent协议完整的操作流程如图3所示。

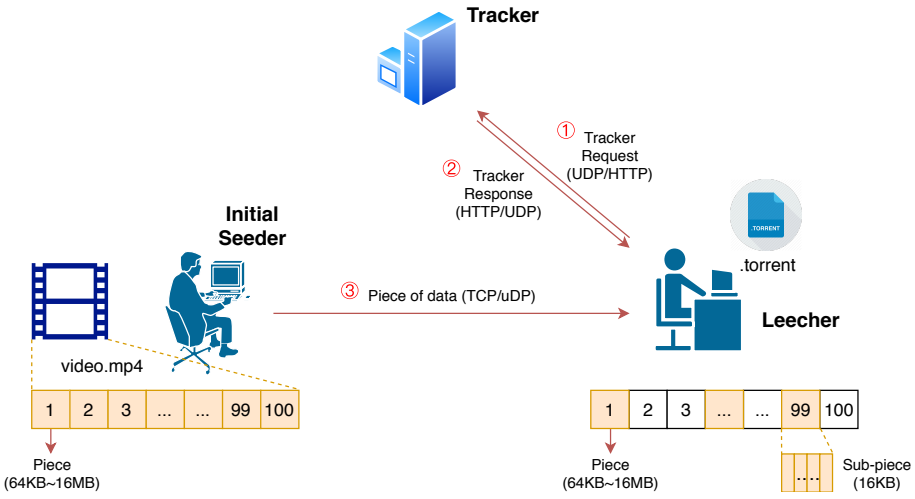

图 3: BitTorrent协议完整流程

初始的播种者将要传输的文件分割为很多片段，每个片段的长度取值可以从64KB到16MB。吸血者从元信息文件的announce键值中获取追踪者的信息，然后通过HTTP或UDP的方式与追踪者建立连接。追踪者将播种者的IP地址、端口号等信息回传给吸血者。吸血者根据对应的信息通过TCP或μDP方式，与播种者建立连接，并逐次下载对应片段。每个片段又可划分为很多个子片段，其中16KB的子片段为最小传输单位。当吸血者获取了所有片段并依序整合后，传输结束。

而吸血者在获取了部分片段后可以转变成播种者，同样共享资源给其他吸血者下载。如图4所示，新的吸血者同样从追踪者处获取播种者名单。追随者默认会返回50个随机的播种者，吸血者将并行地从这些播种者处获取缺失的片段，直到所有片段都收集齐为止。因此BitTorrent协议具有更多播种者，更多资源副本，下载速度更快的特点。

注意，由于种子文件较小，故一般可以通过搜索引擎或专门的种子服务器得到，由于不是本文的重点，故图中未给出种子传播的途径。

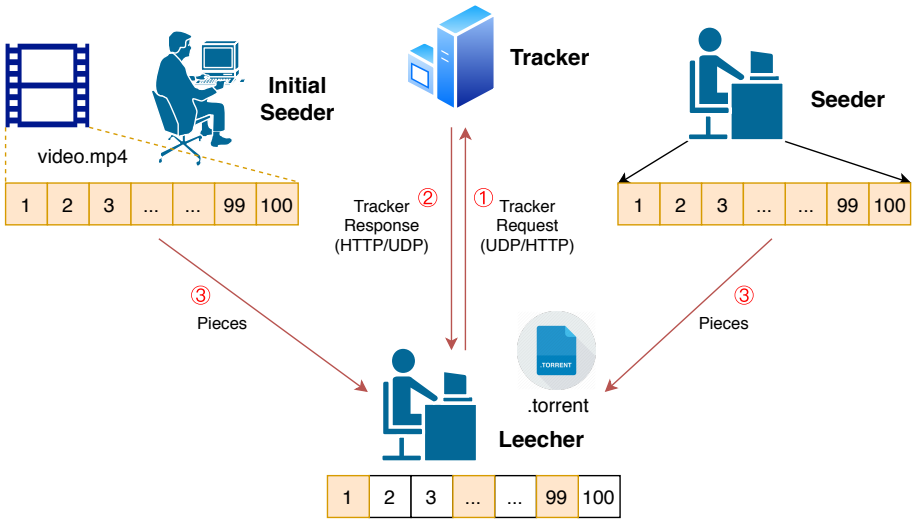

图 4: BitTorrent协议完整流程（多个播种者）

2.7 对等实体选择算法

对等实体选择策略用来决定从哪里下载对应片段，目前BitTorrent协议主要采用以下几种算法 [17]：

- 以牙还牙(Tit-for-tat, TFT)策略：一个对等实体更倾向于将其片段分享给那些提供给它更大下载速率的实体。这个策略大幅提升了BitTorrent协议的性能，有效避免搭便车的实体(free-rider)，即只下载不上传。
- 最优非阻塞：每隔30秒，一个对等实体会随机将它的邻居实体取消阻塞，而无关其上传速率。这能有效发现一些上传速度更快的对等实体。
- 反冷落(anti-snubbing)算法：如果一个实体注意到隔了一段时间它都没有从某一实体处获得片段，则该实体会认为被对方冷落了，并采取反制措施，不再向其上传片段。
- 只上传算法：一旦一个对等实体完成整个文件的下载，它将变成播种者。因为播种者没有其他东西要下载，因此无法基于下载速率选择最好的对等实体。相反地，播种者会更

倾向于选择与有更优上传速率的实体进行分享。

2.8 片段选择算法

由于吸血者需要从播种者处获取文件片段(piece)，最后再一并整合，故获取哪一个片段是需要考虑的问题，常见的有以下几种算法：

- 严格优先：对等实体只关注于当前下载的片段，先将当前片段的所有子片段下载完，才下载其他片段。这可以保证自己有完整的片段可以与其他实体进行交易。
- 稀少优先：这是一般的情况，对于在所有对等实体中最缺失的片段，应该最先下载。这可以保证最常见的片段留到最后才进行下载。
- 随机优先：同字面意思，随机选取片段进行下载。
- 游戏终止模式(End game mode)：为避免获取最后几个片段造成的延迟，对等实体将向其他所有实体发送请求。而一旦片段获得，则应发送`cancel`消息给不必要的对等实体，防止带宽浪费。

3 实现细节

4 总结

本文简要介绍了对等网络的特点，并以BitTorrent协议为例进行深入研究。通过对BitTorrent协议文档的整理，阐述该协议的一些基本概念以及工作原理。同时给出具体的实现方法，给后面的研究者提供了一定的参考。

参考文献

[1] Wikipedia. Peer-to-peer. https://en.wikipedia.org/wiki/Peer-to-peer.

[2] Tencent. QQ live. http://live.qq.com/.

[3] Bitcoin. Bitcoin. https://bitcoin.org/en/.

[4] Blockchain. Blockchain. https://www.blockchain.com/.

[5] Matteo Gianpietro Zago. Why the web 3.0 matters and you should know about it. https://medium.com/@matteozago/why-the-web-3-0-matters-and-you-should-know-about-it-a5851d63c949, 2018.

[6] Wikipedia. Bittorrent. https://en.wikipedia.org/wiki/BitTorrent.

[7] ZDNet. Cachelogic says 35 https://www.zdnet.com/article/cachelogic-says-35-of-all-internet-traffic-is-now-bittorrent/.

[8] Ernesto. Thunder blasts utorrent's market share away. https://web.archive.org/web/20160220105711/https://torrentfreak.com/thunder-blasts-utorrents-market-share-away-091204/, 2009.

[9] uTorrent. utorrent. https://www.utorrent.com/intl/zh/.

[10] Facebook. Facebook. https://www.facebook.com/.

[11] Twitter. Twitter. https://twitter.com/?lang=en.

[12] Digital Forensics Community in Korea. Understanding of bittorrent protocol. http://dandylife.net/docs/BitTorrent-Protocol.pdf.

[13] Bram Cohen. Bep 3 - the bittorrent protocol specification. http://bittorrent.org/beps/bep_0003.html, 2017.

[14] Arvid Norberg. Bep 29 - utorrent transport protocol. http://bittorrent.org/beps/bep_0029.html, 2017.

[15] Olaf van der Spek. Bep 15 - udp tracker protocol for bittorrent. http://bittorrent.org/beps/bep_0015.html, 2017.

[16] Andrew Loewenstern and Arvid Norberg. Bep 5 - dht protocol. http://bittorrent.org/beps/bep_0005.html, 2017.

[17] Raymond Lei Xia and Jogesh K. Muppala. A survey of bittorrent performance, 2010.