An Overview of Media Streams Caching in Peer-to-Peer Systems

WAHEED YASIN, HAMIDAH IBRAHIM*, NUR IZURA UDZIR AND NOR ASILAH WATI ABDUL HAMID

Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 UPM, Serdang, Selangor Darul Ehsan, Malaysia
*Corresponding author: hamidah@fsktm.upm.edu.my

Nowadays, the idea of media contents streaming through the Internet has become a very important issue. On the other hand, full caching for media objects is not a practical solution and leads to consume the cache storage in keeping few media objects because of its limited capacity. Furthermore, repeated traffic which is being sent to clients wastes the network bandwidth. Thus, utilizing the bandwidth of the network is considered as an important objective for network administrators. Media objects have some characteristics that have to be considered when a caching algorithm is going to be performed. In this paper, recent approaches that have been proposed for media streams caching in peer-to-peer systems are reviewed.

Keywords: media streaming; caching; peer-to-peer systems; proxy servers

Received 4 July 2012; revised 20 April 2013
Handling editor: Suchi Bhandarkar

1. INTRODUCTION

Utilizing the bandwidth of a network is considered as an important aim for network administrators [1]. On the other hand, repeated traffic that is being sent to clients wastes the bandwidth of the network, which might lead to the following:

(i) More delays for web users due to the increase in traffic on the Internet [2, 3].
(ii) Push network administrators to extend networks’ gateways at rates that are out of proportion to the end-users’ growth [4]. These extensions of bandwidth are immediately consumed without reaching the expected improvement of network performance, which makes web users unsatisfied.

One of the popular approaches that has been adopted is caching. On the other hand, full caching for the media object is not a practical solution and leads to consume cache storage in keeping few media objects because of its limited capacity.

Thus, the aims of this paper are to present media streams caching characteristics and, in addition, to review and classify recent approaches for media streams caching in peer-to-peer systems since there exists no single in-depth survey of media streams caching in peer-to-peer systems. The organization of the article is as follows: Section 2 presents the benefits and limitations of caching. Section 3 presents the characteristics of media streaming. Section 4 presents media streaming systems. Section 5 presents the major policies of partial caching of media objects. Section 6 presents a transcoding proxy. Section 7 presents peer-to-peer systems, while Section 8 presents peer-to-peer media streams techniques. Section 9 reports some of recent research work related to this overview. Finally, Section 10 presents the conclusion of this work.

2. CACHING

In computer engineering, caching is the process of storing data in an intermediate medium called a cache and it is used for serving future queries instead of fetching data from original sources. A cache hit indicates that the system answers the query from the data stored in the cache, while a cache miss indicates that the system does not have the data. Several benefits can be drawn from applying caching such as reducing the system load, the latency and the bandwidth consumption. In this section, caching stages and limitations are presented.
2.1. Caching stages

Media streams caching might be performed in different stages [3], which can be listed as follows:

(i) **Browser stage.** In this level, caching is performed close to the web user such as caching using a computer’s hard disk.

(ii) **Proxy stage.** Proxy servers located at the network edges in client–server systems. Proxy servers can store files and directly serve users’ requests in the network. Thus, it reduces the traffic that needs to be transmitted across the network.

(iii) **Web server stage.** In this stage, caching is performed at the source of web content that is usually at the network of Internet service providers, which is a web cache that is shared among all users of that network.

2.2. Caching limitations

Although there are a lot of benefits that a cache can provide, caching has limitations that have to be taken into account when a cache is deployed in the system. Some of these limitations are discussed in this section.

(i) **Data consistency.** The consistency of data being saved in the cache is one of the most important issues that should be considered. Web contents might be updated in the original servers which results in data invalidity in the cache. Thus, when there is a request for a datum which has just been updated, the cache has to be informed about the new updates. Many researches have been conducted in the area of cache consistency [5].

(ii) **Web caching security.** Web caching is good for convenience and performance but there is a security cost for user convenience because cached objects might be vulnerable to some threats. For example, disclosure of unauthorized data via accessing cached objects and increasing of user impersonation and privileges via cookies and cached sessions. Thus, data privacy is an important issue that should be considered in caching. That means data that are considered sensitive should be maintained in a different way. One technique might be used is to categorize data into two groups: non-cacheable data and cacheable data. Non-cacheable data are data that are considered as sensitive data; while cacheable data are data that are not considered as sensitive as the previous category [5]. Thus, the cache might only keep cacheable data in its storage.

(iii) **Data replacement.** A replacement mechanism is a mechanism that refers to the process that happens when the cache becomes full and there is not enough space for new items, which leads to the task of removing old items to make space for new ones. Thus, the replacement mechanism has to decide which items are needed to be stored, and which items have to be removed. Many factors should be considered when the cache has to take a replacement decision [6]. Some of these factors are listed below.

(a) **Recency:** Time since the last use of a web object.
(b) **Frequency:** Total number of requests of a web object.
(c) **Cost of fetching a web object:** Cost to fetch a web object from the original source, including processing, bandwidth and other resources.
(d) **Modification time:** The time since the last modification.
(e) **Expiration time:** The time a web object becomes useless and needs to be replaced.

A simple mechanism to perform data replacement might be based on a First In First Out approach. That means replacing the first coming web item in the queue of the web by the new one. On the other hand, least recently used (LRU) is one of the common mechanisms that is used to perform data replacement. The basic approach of LRU is to remove first the web items that are LRU from the storage of the cache. Thus, in order to perform that, the LRU mechanism has to track these web items by implementing a recency index for each item, which indicates the time since the last use of the item. The least used one will be replaced with the new coming item. Many variants of LRU have been proposed in the literature such as SVM-LRU and NB-LRU [7, 8].

Another common data replacement mechanism is called least frequently used (LFU), which its basic approach of which is to remove first the web items that are LFU from the cache. Thus, the LFU mechanism has to track these web items by implementing a frequency index for each item, which indicates the total number of requests. A web item with the least index value will be replaced with a new coming item. Many problems occur when the LFU mechanism is implemented in the cache. For example, a new web item might likely be accessed again in the near future. However, the value of its frequency index is low if it is compared with the frequency indices of other cached web items which forces the LFU mechanism to replace it with a new coming web item. Many variants of LFU have been proposed in the literature such as LFU-Aging and LFU-DA [6, 7].

3. MEDIA STREAMING

Media streaming characteristics have been addressed in [9, 10]. In this section, the characteristics of media streaming are presented.

(i) **Media object size.** Media objects have a huge size because of their data volume. For example, a media object such as a video file might have a size of more than 100 MB, while a web page might have only 100 KB. Thus, keeping the entire file will waste the cache storage.
The characteristics of some media streaming systems and their limitations are discussed below.

4. MEDIA STREAMING SYSTEMS

The characteristics of some media streaming systems and their limitations are discussed below.

(i) Media streaming proxy systems. Media objects can be delivered through proxy systems; however, the restriction of cache space remains, in addition to restricted bandwidth which is not able to deliver media objects to clients simultaneously, which is considered as a scalability problem. Thus, a key limitation of streaming media objects through proxy systems is that a proxy cache may become a system bottleneck that forms a single point of failure [11].

(ii) Content delivery networks (CDNs). To move media objects close to clients, a content delivery/distribution network (CDN) has been deployed in the literature. For example, (AKAMAI) http://www.akamai.com and portal infrastructure for streaming media presented in [12] are CDN architectures for streaming media objects with high quality over the Internet. A CDN has to place multiple copies of media contents or sometimes part of them on its infrastructure. On the other hand, a CDN is considered as an expensive solution for small media providers because of the huge number of servers within the network.

(iii) Peer-to-peer streaming systems. A peer-to-peer streaming system has been proposed to utilize peers’ resources in a decentralized manner such as PROMISE as presented in [13]. In the peer-to-peer approach, a peer works both as a client and a server. The Peer-to-peer approach has also some limitations in providing a good level of QoS in a high dynamic network that might not be satisfactory for end-users. Furthermore, each peer maintains its cache separately and no coordination of media contents is available between peers. Thus, the availability and reliability of the streaming service might not be ensured.

(iv) Peer-to-peer assisted streaming proxy systems. To overcome the scalability problem in the proxy-based streaming systems, and the problem of availability and reliability of the streaming service in peer-to-peer systems, peer-to-peer assisted proxy systems have been proposed such as PROP [11]. A key limitation of the PROP system is that only one streaming server is working at a certain time.

5. PARTIAL CACHING OF MEDIA OBJECTS

Full caching of media objects is not a practical solution and leads to consumption of cache storage in storing a few media objects because of its limited capacity. On the other hand, the media object content is rarely updated which results in management issues like cache consistency and coherence being less critical. There are two major policies that have been proposed in the literature for partial caching of media objects, namely: time-based partial caching and bandwidth-based partial caching or sometimes called space-based partial caching [10].

5.1. Time-based partial caching

Time-based partial caching assumes that a media object is fragmented in time, where each media fragment is called a segment. For example, a 60 min movie might be fragmented into three segments, the first 2 min, the next 10 min and the remaining 48 min, respectively. Basically, two techniques have adopted the time-based partial caching for caching segments that are Prefix Caching and Caching of Arbitrary Segments [10].

(i) Prefix caching. Prefix caching is a time-based partial caching, where a media file is fragmented into two segments, namely the prefix and the suffix. The prefix segment represents the initial part of the media object, while the suffix represents the rest of this media object. In prefix caching, proxies only cache the prefix segment.
of a media object, while the suffix remains in the original server. As soon as a proxy receives a request for a media object from a client, it starts streaming the prefix to the client, provided that it is cached in the proxy. At the same time, it fetches the suffix from the origin server. The motivation of prefix caching is that end-users are likely to view the initial part of a media object more than the latter parts. On the other hand, the size of the prefix is considered as a key factor in prefix caching.

(ii) Caching of arbitrary segments. In this type of caching, each segment is double the size of the previous segment. Thus, the segment’s size is exponentially increased. Assume that a 75 min movie is exponentially fragmented into four segments: Segment 1, Segment 2, Segment 3 and Segment 4. That means if the duration of Segment 1 is 5 min, the durations of the remaining segments will be 10, 20 and 40 min, respectively. Using unequal sizes of segments has some advantages. The initial segments are allocated small sizes because end-users are likely to view the initial part of a media object more than the latter parts. Moreover, using unequal sizes of segments makes the cache space free faster by disposing of the large sizes of segments when these segments become unpopular.

Both, caching of arbitrary segments and prefix caching have been combined in one policy, which is called selective caching which allows end-user interaction operations.

5.2. Bandwidth-based partial caching

In this policy, the media object is fragmented in space (bandwidth dimension). That means each media object is fragmented into a base layer and subsequent enhancement layers as illustrated in Fig. 1.

Basically, a media object is fragmented into $n$ layers, the base layer or Layer 1 which contains a low resolution script of each frame, and $(n - 1)$ enhancement layers that contain additional information to refine the resolution of the media object. Moreover, layered encoding provides diverse resolution presentations of media object that plays a key role in heterogeneous systems. Many techniques have adopted bandwidth partial caching policy. Two basic techniques are presented in this article, namely: video staging and caching of layered video [10].

(i) Video staging. In the video staging technique, a media object is divided into two layers that are a fixed rate layer and a variable rate layer. The lower layer is stored at the origin server, while the upper layer is cached at the proxy. As soon as a proxy receives a request for a media object from a client, it starts streaming the variable rate layer to the client, while the fixed rate layer is fetched from the origin server. Thus, it reduces the backbone bandwidth requirement. It can be observed that video staging imposes a fixed overhead on both of the origin server and the network path from the server to the proxy as a result of the fixed rate of the lower layer.

(ii) Caching of layered video. This technique assumes layered media objects and caches different layers at a proxy, while fetching others from the origin server based on demand principles. Moreover, caching of layered video may use a rate adaptive protocol for prefetching missing layers in order to improve the quality of cached media.

6. TRANSCODING PROXY

A transcoding proxy is used in heterogeneous environments where clients such as mobile devices, laptops, etc. have different capabilities and different bandwidth capacities. Thus, the transcoding proxy is responsible to deliver a web object and accommodates clients’ requirements. A principle approach of the transcoding proxy is to satisfy a common rule which is based on WhatYou Need Is WhatYou Get media services, and delivers media to users with the quality they expected. This subject is a result of the variability of clients such as platforms, computing power, display, storage and network bandwidth. A transcoding proxy may adapt a media object within media types such as resizing and colour resolution, or between media types such as changing from video type to image type. On the other hand, producing a dedicated media object according to devices’ capabilities results in additional loads on content management [14].

7. PEER-TO-PEER SYSTEMS

There are two different ways of configuring a peer-to-peer system that are structured peer-to-peer systems and unstructured peer-to-peer systems [15]. This section presents the configuration of peer-to-peer systems.

7.1. Unstructured peer-to-peer systems

In unstructured peer-to-peer systems, preexisting infrastructures are not required such as Gnutella [16] and mobile ad
FIGURE 2. A peer-to-peer overlay running on a MANET.

FIGURE 3. Peer-to-peer-based media streaming.

7.2. Structured-based peer-to-peer systems

A structured-based peer-to-peer system needs a preexisting infrastructure such as the availability of access points. Furthermore, each peer keeps track of the resources offered by its neighbours. Thus, fewer messages are needed for answering queries that result in low latency and load balancing. A structured-based peer-to-peer system may employ a distributed hash table (DHT) for object placement and lookup service, in which hash functions are used to map peers and shared content references. Each object is located based on its key value, and it is reached by routing query among nodes according to the DHT routing controls. The DHT stores the tuple (key, value), where key is the media segment identifier and value is the segment index. Several protocols have been proposed for the structured peer-to-peer system such as Pastry [19], CAN [20] and Chord [21]; however, these protocols may not be reliable in a wireless environment, due to some reasons such as high failure rates, excessive overheads and mobility [22].

7.3. Advantages of cooperative caching based on peer-to-peer systems

Many researches have studied the advantages of applying cooperative caching based on peer-to-peer that is called a peer-to-peer assisted proxy streaming system such as in [11, 23, 24]. Some of the peer-to-peer streaming cooperative caching advantages listed in [25] are as follows:

(i) It is considered as an easy-to-apply caching strategy.
(ii) Data forwarding is not restricted to particular directions; however, it is determined dynamically as data are available.
(iii) It is considered as a robust and resilient caching strategy.

For example, assume that a file ‘bestvideo.avi’ is requested by a peer A, and this file is available on peers B, C and D. Thus, it is better to download file segments from these peers for many reasons such as: the time that is consumed to download this file is normally less than the time that is consumed to download the file from the proxy. In other words, it can be said that peers work as lightweight proxies in the system. Figure 3 illustrates the peer-to-peer-based media streaming. An advantage of applying peer-to-peer cooperative caching is resource utilization on the peers such as CPU and disk storage that overcomes the limitation of the proxy storage capacity.

8. PEER-TO-PEER STREAMING TECHNIQUES

In peer-to-peer streaming, a peer or a client may receive a stream from the origin servers, proxy caches or other peers. On the other hand, peers may form a kind of multicast trees; however, these trees are unreliable because a peer may join and leave at any time. Thus, a peer-to-peer streaming technique must adapt multicast trees according to the changes including peers’ status and other network parameters. Moreover, it has to
employ a backup streaming technique to overcome interruptions and packet losses caused by peer’s joining and leaving [10].

In this section, we go through some of the recent peer-to-peer streaming techniques.

8.1. Zigzag streaming

Zigzag streaming is a peer-to-peer approach that aims to stream media objects to a large number of peers based on tree construction algorithms. Tree construction in zigzag streaming assumes a logarithmic growth of the tree’s depth with the number of peers. Thus, a smaller depth means a small number of intermediate peers between the origin server and the client, which results in reducing end-to-end delay. The number of children of each peer in the proposed tree is bounded, which limits bandwidth requirements at each peer. Furthermore, children of any peer are reconnected to other peers in the network, as soon as it leaves the network. On the other hand, the drawbacks of zigzag streaming appear in routing and slow convergence [26].

8.2. Layered peer-to-peer streaming

In [27], a peer-to-peer streaming technique based on layered encoding has been proposed. This technique assumes that each media object is encoded in separate layers. Furthermore, it assumes a heterogeneous environment. A layered peer-to-peer streaming technique employs an algorithm that determines layers to be fetched by a peer from upstream peers and layers that should be cached to serve downstream peers. This algorithm takes into account the available bandwidth in both directions, upstream and downstream, when QoS has to be considered. For example, if a peer is caching five layers of a media object, it may stream only three layers from the media object because its child peer does not have available bandwidth.

8.3. SplitStream

In [28], the SplitStream technique has been proposed for peer-to-peer streaming, where a media object is divided into stripes that are distributed through several multicast trees. SplitStream applies application-level multicast that forms a forest of multicast trees, where each peer is considered as a node in one multicast tree and, at the same time, as a leaf of other multicast trees. SplitStream applies a layered coding method for media object encoding, as each layer can be distributed through a different multicast tree. SplitStream is DHT-based, which is difficult to implement because of its high complexity.

8.4. Cooperative networking resilient peer-to-peer streaming

Cooperative Networking (CoopNet) is a resilient peer-to-peer streaming technique that overcomes disruptions and packet losses caused by peers joining and leaving by using redundant network paths and data transmissions [28]. CoopNet employs multiple description coding (MDC), which is a method of encoding a media stream into M sub-streams or descriptors. The number of descriptors received at the client side determines the resolution of the reconstructed stream. MDC differs from layered coding in that every subset of descriptors must be decodable, while in layered coding only a nested sequence of subsets must be decodable. In CoopNet technique, descriptors are sent into different paths that ensure that clients are able to reconstruct the media object since they receive subsets of the descriptors. CoopNet is centralized, which makes it vulnerable to a single point of failure. Moreover, CoopNet is not scalable.

8.5. Collaborative peer-to-peer streaming for heterogeneous environments

In [29], a technique based on a peer-to-peer system has been proposed for media streams in heterogeneous environments where network terminals have distinct characteristics (e.g. bandwidth and level of QoS). This technique is centric-receiver and utilizes media coding to increase the throughput and reduce the packet drop ratio in order to improve the overall QoS.

8.6. PROMISE peer-to-peer streaming

To judiciously construct a peer-to-peer tree, a peer-to-peer streaming technique called PROMISE has been proposed in [13]. It supports a service to deduce the underlying network topology between peers that may help in judiciously constructing a broadcasting tree where paths between two interconnected peers are short. Furthermore, it monitors the status of peers as to whether they are still connected to the network, and detects departures of peers with low latency. A CollectCast is a multicast protocol that is performed in the system in order to find the best peers for media streaming. PROMISE also employs a service for failure recovery which changes the status of peers to standby as their parents leave the network. PROMISE is DHT-based, which is difficult to implement because of its high complexity.

8.7. Peer-to-peer streaming based on network coding

In [30], a peer-to-peer media streaming technique based on network coding has been proposed. The network coding in this technique is used to reduce the probability of node failure. The proposed technique optimizes the node selection policy in order to reduce media transmission delay, network latency and the packet loss ratio; however, it adds computational complexity to the system.
### TABLE 1. Peer-to-peer streaming techniques.

<table>
<thead>
<tr>
<th>Streaming technique</th>
<th>Brief description</th>
<th>Features</th>
<th>Limitations</th>
<th>continued.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zigzag streaming [26]</td>
<td>It is based on tree construction and assumes a logarithmic growth of the tree’s depth with the number of peers.</td>
<td>It limits bandwidth requirements at each peer</td>
<td>It is a slow convergence technique and has a routing problem</td>
<td></td>
</tr>
<tr>
<td>Layered peer-to-peer [27]</td>
<td>It assumes that each media object is encoded in separate layers. Furthermore, it employs an algorithm which determines layers to be fetched by a peer from upstream peers and layers that should be cached to serve downstream peers.</td>
<td>It takes into account the available bandwidth in both directions, upstream and downstream</td>
<td>In layered coding only a nested sequence of subsets must be decodable</td>
<td></td>
</tr>
<tr>
<td>SplitStream [28]</td>
<td>A media object is divided into stripes that are distributed through several multicast trees. Also, it applies an application-level multicast that forms a forest of multicast trees.</td>
<td>Each layer can be distributed through a different multicast tree</td>
<td>It is a DHT-based technique which is difficult to implement because of its high complexity</td>
<td></td>
</tr>
<tr>
<td>CoopNet [28]</td>
<td>It employs MDC, which is a method of encoding a media stream into $M$ sub-streams or descriptors. The number of descriptors received at the client side determines the resolution of the reconstructed stream.</td>
<td>Descriptors are sent into different paths that ensure the clients are able to reconstruct the media object</td>
<td>It is centralized, which makes it vulnerable to a single point of failure. Moreover, it is not scalable</td>
<td></td>
</tr>
<tr>
<td>Collaborative peer-to-peer streaming for heterogeneous environments [29]</td>
<td>It is proposed for media streams in heterogeneous environments where network terminals have distinct characteristics (e.g. bandwidth and level of QoS)</td>
<td>Increases the throughput and reduces the packet drop ratio in order to improve the overall QoS</td>
<td>It adds an overhead and suffers from fluctuation in the received video</td>
<td></td>
</tr>
<tr>
<td>PROMISE [13]</td>
<td>It supports a service to deduce the underlying network topology between peers that may help in judiciously constructing a broadcasting tree where paths between two interconnected peers are short.</td>
<td>It employs a service for failure recovery. Also, it monitors the status of peers’ connectivity and detects departures of peers with low latency</td>
<td>It is a DHT-based technique which is difficult to implement because of its high complexity</td>
<td></td>
</tr>
<tr>
<td>Based on network coding [30]</td>
<td>The network coding in this technique is used to reduce the probability of node failure and optimize the node selection policy.</td>
<td>It reduces the media transmission delay, network latency and the packet loss ratio</td>
<td>It adds computational complexity to the system</td>
<td></td>
</tr>
<tr>
<td>Zhang-Tan-2010 [31]</td>
<td>It uses nodes’ access rate during a certain period of time to choose a set of candidate caching nodes. Furthermore, it uses a caching strategy to determine the size of data caching buffer according to nodes’ I/O bandwidth and the network’s bandwidth.</td>
<td>It reduces the number of query messages and data delays</td>
<td>It uses message passing technique, which has a complex communication structure</td>
<td></td>
</tr>
</tbody>
</table>
**TABLE 1.** continued.

<table>
<thead>
<tr>
<th>Streaming technique</th>
<th>Brief description</th>
<th>Features</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on hybrid push–pull [32]</td>
<td>A hybrid push–pull approach has been adopted for exchanging media objects</td>
<td>It reduces mesh delays in peer-to-peer systems</td>
<td>It increases the traffic overhead of the system. Also, there is no guarantee of finding a supplier node in time.</td>
</tr>
<tr>
<td>CLive [33]</td>
<td>It employs cloud computing infrastructure to assist the unstructured peer-to-peer streaming system and uses a mesh-pull approach for exchanging media objects</td>
<td>It improves the QoS of media streams</td>
<td>It suffers from the high cost of rental cloud resources</td>
</tr>
</tbody>
</table>

8.8. **Zhang-Tan-2010 peer-to-peer media streaming technique**

In [31], a caching technique for streaming media data has been proposed based on peer-to-peer systems using the access rate of nodes during a certain period of time to choose a set of candidate caching nodes. This technique is called Zhang-Tan-2010 in our paper. Zhang-Tan-2010 technique uses a local caching strategy to determine the size of the data caching buffer according to nodes’ I/O bandwidth and the network’s bandwidth. In this technique, the nodes of the peer-to-peer system are divided into three categories: source nodes that provide streaming media files, caching nodes that cache media data partly and request nodes that initiate requests for media data. The Zhang-Tan-2010 technique reduces the number of query messages and the load of the network effectively. Also, it reduces the data delays. Thus, the whole system’s performance is improved.

8.9. **Peer-to-peer streaming based on hybrid push and pull approaches**

A hybrid push–pull peer-to-peer media streaming technique has been proposed in [32]. This technique applies a mesh topology to select the most optimal media segment to be pushed and pulled in a parallel fashion using buffermap exchange information. The hybrid technique reduces mesh delays in peer-to-peer systems; however, it increases the traffic overhead of the system. Moreover, there is no guarantee to find a supplier node in time.

8.10. **CLive peer-to-peer streaming technique**

To improve the QoS of media streams, CLive has been proposed in [33]. CLive employs a cloud computing infrastructure to assist the peer-to-peer streaming overlay in order to guarantee a predefined level of QoS with the lowest cost. Peers in CLive are configured as an unstructured peer-to-peer system and mesh-pull approach adopted for exchanging media objects. CLive adds cloud resources that are called helpers to increase the available bandwidth and the probability of receiving media objects on time. CLive suffers from the high cost of rental cloud resources.

Table 1 summarizes the existing peer-to-peer media stream caching techniques and their limitations.

9. **RELATED WORKS**

In this section, we report some of recent researches on utilizing the network bandwidth based on the caching strategy in peer-to-peer systems.

In [34], caching has been adopted to reduce the overhead imposed by flooding. First, a principle called temporal locality has to be understood, its idea being that data which have been recently requested are requested again. The proposed approach is called interest-based shortcuts and it relies on the principle of interest-based locality where peers are organized logically according to their interests over the existing network. The assumption of interest-based locality is that a peer that has a media object that other peers are interested in may have other media objects that they are also interested in. Thus, if a peer’s request exhibits a cache miss, it is forwarded to another peer that has the same interests. It has been concluded that keeping query results in the cache reduces the overhead and network traffic that is imposed without caching.

A static caching algorithm is a way of caching where the most frequently requested items are stored in the cache. This algorithm may work optimally if web items have the same size. Furthermore, requests are independent and identically distributed. In [1], it has been shown that the static caching algorithm still performs close to the optimal results for large sizes such as media items, when requests are strongly correlated.

In [35], an admission-control technique called screening has been proposed to improve the proxy caching performance by reducing the delay that is experienced by web users. This technique can be applied in the web server and proxy stages.
Screening classifies media objects into two categories, which are cacheable and non-cacheable based on loading times, and it uses the LRU algorithm for caching web items. The variation in media object size has a negative impact on the performance of screening, which is considered as a drawback.

In [2], algorithms for implementing a network of nodes’ caches have been developed for centralized and decentralized environments in order to address the problem of wasting storage space by storing unnecessary repeated objects in caches without taking into account that these objects have already been cached by neighbouring nodes. An experiment has been performed, and its results show that the implementation of these proposed algorithms improve the performance of a network when nodes take into account media objects that have already been cached in their neighbours’ caches.

In [36], a scalable proxy caching algorithm has been proposed in order to minimize the channel bandwidth and the buffer size of a client. This algorithm is based on the fact that changing a video frame depends on the relative frame position in the time axis as well as the frame size. An experiment has been performed, and its results show that the performance of the proposed algorithm is better than selective caching for QoS networks and Prefix algorithms [37].

In [9], a study on the properties of peer-to-peer live streaming data requests has been conducted, which shows that peer-to-peer live streaming traffic can be answered from the cache. Also, a caching algorithm that is based on a sliding window has been proposed, called an SLW algorithm, which has been evaluated and compared with traditional caching strategies. An experiment has been performed, and its results demonstrate that the SLW algorithm performs close to an off-line optimal algorithm which has full knowledge of future requests.

A peer-to-peer system for media stream caching is one of the most challenging peer-to-peer applications since it is vulnerable to many attacks. In [38], a survey on privacy and security concerns in peer-to-peer systems has been presented, which addresses the common attacks.

In [39], a prefetching caching, or sometimes called proactive caching, protocol has been proposed to be applied at the client side. This proposed protocol uses a threshold that might be set dynamically in order to optimize prefetching gain. Although prefetching systems improve the responsiveness for web users, they might increase network traffic load caused by the inaccuracy of prediction algorithms. Prefetching caching protocols might succeed in special cases under certain circumstances. Furthermore, the variation in the size of a media object has negative effects on the performance of this approach.

In [40], the problem of determining the appropriate variant of a media object has been addressed in addition to which variant has to be cached at a particular node in a network. Furthermore, a model for caching media objects has been proposed by considering both the transcoding and transmission costs. Also, an algorithm to determine the optimized location has been presented. The proposed algorithm is based on symmetric routing paths only; thus, it might not work properly in asymmetric cases.

One of the recent caching strategies has been presented in [23], which proposed cooperative caching by overlying a peer-to-peer network on a client–server network for push-based broadcast. This strategy also has some limitations such as each node has to know the broadcast programme. Moreover, the signalling overhead is ignorable such as the time and the bandwidth consumed by query and reply messages.

In [8, 41], new approaches based on machine learning techniques have been proposed to improve the efficiency of traditional cache replacement strategies such as LRU, Greedy-Dual-Size and LFU. However, these new approaches have some limitations in the additional computational overhead that is required in the preparation of the target outputs in the training phase when looking for future requests.

10. CONCLUSION AND FUTURE WORK

Media objects web caching improves the performance of media streams systems in terms of delivery time. In this work, some of the recent approaches for media streams caching in peer-to-peer systems have been reviewed; in addition, the characteristics of media caching algorithms have been presented.

Many peer-to-peer streaming research techniques have been proposed in the literature. Most of them have focused on the topics listed below:

(i) Tree construction algorithms.
(ii) Network maintenance when peers join or leave the network.
(iii) Back-up streaming techniques that overcome interruptions and packet losses caused by peers joining and leaving.
(iv) The QoS of media streams.

As a result, media streams caching adds some advantages to the cache proxy as listed below:

(i) Media objects redundancy. Owing to media objects’ replications cache proxies can continue delivering media contents to end-users in case any transit failure occurs at the origin server.
(ii) Reducing start-up latencies for accessing media objects.
(iii) Back-up streaming techniques that overcome interruptions and packet losses caused by peers joining and leaving. Cache proxies are located close to end-users, which results in reducing start-up latencies imposed for accessing media objects.
(iv) Reducing original server loads by distributing them on different cache proxies.

Furthermore, the area of media streaming would continue focusing on streams caching in different stages with different devices such as personal digital assistants and mobile phones.
Our plan for future work in this topic is to show the effects of media streams caching on the performance of the network in different stages.

REFERENCES


