A standalone content sharing application for spontaneous communities of mobile handhelds* †

Amir KRIFA, Mohamed Karim SBAI, Chadi BARAKAT, Thierry TURLETTI
Planète Project-team, INRIA, France
{akrifa, mksbai, cbarakat, turletti}@sophia.inria.fr

ABSTRACT

This demo illustrates the benefits of BitHoc, a standalone protocol for content sharing among spontaneous communities of mobile handhelds using wireless multi-hop connections. BitHoc is a Trackerless BitTorrent-like application adapted to mobile wireless ad-hoc networks(MANET). The current BitHoc architecture is composed of three principal components: a content sharing service, a membership management service and a content discovery service. The present demo highlights the efficiency of the BitHoc package in dealing with diverse challenges encountered in the MANET environment. Our solution considers the following issues: routing overhead, sharing opportunities and mobility of nodes. In order to validate the feasibility of our application and evaluate its performance, we consider a test-bed composed of PDAs and smartphones equipped with WIFI adapters and Windows Mobile 6 operating system.

1. INTRODUCTION

Content sharing is currently a universal concern among computer users and has become soon an important requirement for mobile handheld users. In fact, nowadays, the market proposes a variety of mobile devices offering user-friendly interfaces, long-life battery autonomy, sufficient computational power and efficient wireless connectivity. This tremendous advancement triggers the necessity of supporting the very fashionable desktop applications in such mobile environments. Indeed, thanks to the efficient wireless connectivity offered by mobile devices (PDAs, smartphones...), users are frequently brought to locate and share content of interest(data, photos, videos, etc) with other members of the same spontaneous community. Nowadays, they are mainly using point-to-point basic connections, which can be considered as an efficient solution when the number of users interested in the sharing session is very small. Instead, in the case of a large community (for example, mobile handheld users assisting to a conference and willing to share some papers), one is facing the following problem: Increasing the number of parallel point-to-point communications may decrease the global ad-hoc network capacity while decreasing them may increase dramatically the download time. The multi-hop point-to-point communication over long paths is also a serious issue. Hence, there is a strong need to organize the communication overlay among nodes in a way to distribute fairly the burden of data sharing among the set of participants while aiming to decrease the global download time. P2P file sharing solutions are good candidates for such infrastructureless networks as they are based on multi-sourcing which balances resources consumption among peers and reduces the dependency on any central entity. But unfortunately, P2P content sharing applications developed for the Internet cannot directly be plugged and used into mobile devices [5][7]. Indeed, on one hand, these algorithms are not adapted to the constraints of multi-hop wireless networks. For example, it is known that in a resource constrained environment, the choice of the peers to whom to connect cannot be done independently of information on the underlying dynamic topology[5]. Moreover, centralized peer management approaches like the tracker used in BitTorrent do not perform well in such environment as the tracker can be either far away or even invisible by some peers because of disconnections. Furthermore, computer users rely on Internet search engines and dedicated desktop applications to look for the content they are willing to share. This approach becomes obsolete in the case of a spontaneous MANET based community and thus, a dedicated distributed content discovery approach must be provided. On the other hand, from a technical point of view, limited Software Development Kits (SDK) proposed for mobile handhelds represent only a subset of classical SDK(s) used for desktops which leads to incompatibility problems.

This demo presents BitHoc, an open-source standalone software solution for content sharing in MANETs that overcomes the challenges aforementioned. Our architecture consists of three main components: a content discovery service (BitHoc search engine), a membership management service (distributed BitHoc tracker) and a content sharing service (BitHoc client[1]). BitHoc tracker agents installed in nodes

*This work has been supported by the ITEA European project on experience sharing in mobile communities (Expeshare).

†Demo description in proceedings of the ACM SIGCOMM MobiHeld Workshop, Barcelona, Aug 2009.

[1]Preliminary versions of our BitHoc Client and Tracker have been presented in [4]. In this demo, we present the full

connect to each other in order to form and manage the per-content sharing overlay needed by the BitHoc client. This emulates the central tracker of standard BitTorrent. They also construct the content discovery overlay used by the BitHoc search engine to enable content publishing and discovery. To connect to a sharing overlay, a node needs to retrieve a torrent file (a meta-data file) by connecting to the discovery overlay. The members of this overlay manage a distributed database that contains for each torrent the list of nodes uploading it and a short description of the related content. To ensure content sharing, the BitHoc client decides using routing information of the structure of the data exchange overlay seen by him (with whom to exchange data). It also manages the scheduling of data pieces among devices. Our main contribution here is to adapt the peer neighbor and piece selection strategies of BitTorrent to account for the topology of the network and for the scarcity and shared nature of resources. The algorithms of BitHoc are detailed in [5].

We developed our software on mobile devices having Windows Mobile 6 as an operating system and equipped with WIFI adapters. As permanent topology information is required by BitHoc, we choose to use the proactive routing protocol OLSR [3]. For the evaluation, we measure the performance of BitHoc in regards to the download time and sharing ratio metrics. Our test-bed allows us to experiment with the different features of our solution and to compare our version of the BitTorrent's algorithms to the ordinary Internet one when deployed in the wireless ad-hoc environment. The performance analysis based on experimentation shows that BitHoc outperforms the standard version.

The remainder of this paper is organized as follows. Section 2 analyses the objectives of BitHoc. Section 3 describes its architecture. Section 4 presents the demo and the test-bed we have used to test and evaluate our solution.

2. REQUIREMENTS ANALYSIS

We designed our package in such a way to be standalone. So, the user has just to install the software to start publishing and discovering contents and sharing them later with those with the same interest. The other nodes collaborate by forwarding packets at the routing level. Through BitHoc, we provided solutions to the following problems:

- In the classical version of BitTorrent [2], peers periodically contact a central rendezvous point called *Tracker* to obtain fresh information about the peers interested in a specific content and to update their information on the progress of the download. This membership information is dynamic since peers can join or leave the content sharing overlay (called torrent) at any time during the session. Because of the inappropriateness and the large overhead of client/server architectures in wireless ad-hoc networks, it is important to introduce a distributed Trackerless solution to manage the membership of the sharing session. The BitHoc tracker component of our architecture is designed for this purpose.
- The classical version of BitTorrent [2] supposes that the cost of sending data packets to peers is in somehow independent of their locations. In an ad-hoc network, performance metrics like achievable throughput, delay, and energy consumption depend strongly on the number of hops to the peer node. So, it is clearly suboptimal and even unrealistic to deal with peers without considering the underlying topology. Furthermore, when applying the classical BitTorrent's incentives in a wireless multi-hop network, nodes fail to reciprocate data fairly among them. The content dissemination scheme is close to a wave transferring data from the initial seed to the farthest peers. Through new peer neighbor selection and content pieces scheduling strategies, our solution is topology-aware and ensures fair sharing. These strategies are described in details in our research work [5].
- To join a sharing session a user should find and download the torrent file related to that session. In the Internet, peers usually find their torrent files by the help of search engines which look for the files in different central servers. This method does not apply in a mobile ad-hoc environment as MANETs. The BitHoc search engine overcomes this challenge by maintaining a distributed torrent files database thanks to the overlay constructed by the BitHoc tracker.

3. BITHOC ARCHITECTURE

Figure 1 depicts the principal components of the BitHoc architecture and the interactions between them. We illustrate these interactions through two typical usage scenarios:

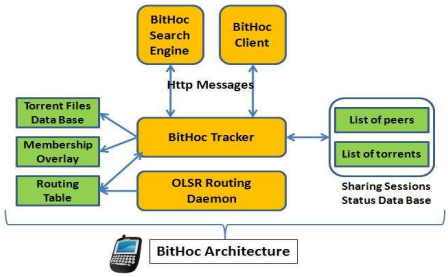

Figure 1: BitHoc Architecture

3.1 Content Publishing and Discovery

A user willing to share some content with the members of his community needs to indicate to the BitHoc client the location of the content in the mobile device file system. First, the client creates a meta-info file (torrent file) that identifies in a unique manner a sharing session of the content. After that, the user publish (locally) the new torrent file and a short text description of the related content using the BitHoc search engine service, which will update the local torrent files database maintained in the underlying BitHoc tracker via HTTP messages. A remote user, willing to share the same content, has to use the BitHoc search engine to find and download the torrent file. He specifies for that the name of the content or some key words related to its description. The request is sent via HTTP messages to its local tracker which looks for the closest match in its local database. If there are no matches, it forwards the HTTP request to the other trackers in the discovery overlay [2]. Then it presents the received results through an ergonomic user

[2]Trackers of the discovery overlay are organized following a dynamic minimum spanning tree that optimizes the dissemination of requests and the retrieval of answers. The management of the overlay members is described in [6].

enhanced content sharing package.

interface (see Figure 2). Based on the details of received answers (fitness to the search, number of peers involved in the sharing session, number of seeders, and number of leechers ...), the user can choose the torrent file to download, then start the content sharing using the BitHoc client.

3.2 Content sharing

Before starting a new sharing session, the user can choose between two versions of BitTorrent algorithms: The classical version [2] and our version adapted to mobile ad-hoc networks [5]. Moreover, the BitHoc client offers a Wizard allowing the user to configure the parameters of BitTorrent (communication ports, choking slot duration, min/max number of peers, etc). Once the torrent file is obtained the BitHoc client can start the sharing session where it can either play the role of a leecher or a seed. It contacts periodically the local BitHoc tracker to get the current list of members of the same content sharing session (torrent) [3]. Using this list and the routing table, it manages the connections with the interested peers. Briefly a client implementing our algorithms exchanges pieces with close peers and only seeds distribute pieces across the network. Note that we allow the user to pause or resume the download while conserving the session context. He can also monitor in real time the status of the session (downloaded bytes, uploaded bytes, numbers of leechers, number of seeders, elapsed time, etc). Furthermore, the BitHoc client keeps in a log file statistics on the content sharing session and provides different levels of event traces. It also manages the storage of the downloaded contents and their classification. Figure 3 shows a screenshot of the BitHoc client.[4]

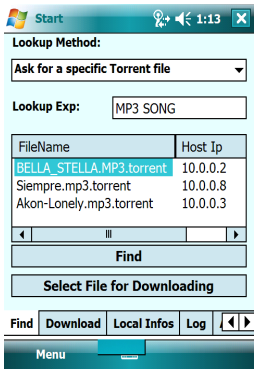

Figure 2: Search Engine

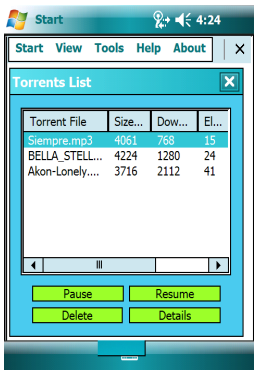

Figure 3: BitHoc Client

4. EXPERIMENTATION AND DEMO

4.1 Test-bed description

Our wireless ad-hoc network experimental environment consists of 14 mobile devices including 7 PDAs (HP iPAQ 214) and 7 smartphones (HP iPAQ 614c). Each handheld is equipped with an IEEE802.11b wireless card. The characteristics of the two types of devices are detailed in[4]. The ad-hoc connectivity is maintained thanks to OLSR daemons run by the different devices. In our experiments, we constructed several network topologies containing a maximum of 6 hops. The objective of the realized swarm is to download a 4 MB MP-3 content. All PDAs were supposed to participate to the sharing of the file. The original seed of the content was chosen randomly among the set of the 14 PDAs. During the demo session, we propose to show and run the same scenario using a subset of these PDAs and also to focus on an on-site evaluation of the BitHoc search engine.

[3]From its side, the BitHoc tracker is in charge of maintaining up-to-date the information on the other peers based on both the events received each time the routing table is modified by the OLSR daemon and the interactions with the other trackers (arrival and departure of peers).

[4]More screenshots of our application are available at[1].

4.2 Some experimentation results

The metrics tracked during our experiments was the download time and average sharing ratio of nodes. We define R_h as the sharing ratio of peers located at h hops from the original seed. It measures the level of reciprocity between downloads and uploads. In the ideal case, the ratio is equal to 1. The two versions of BitTorrent have been tested and the results are presented in Figures 4 and 5. Figure 4 shows a dramatic increase of sharing opportunities when our adapted version is deployed. The routing overhead yielded by the classical version makes any gain obtained by important diversification of pieces negligible. Our method finds the good equilibrium between sharing and diversification. Figure 5 shows that BitHoc outperforms the classical version of BitTorrent in terms of download time. It is accordance to our research results[5]. More information about our research, our experiments and our GPL licensed open-source code can be found on the BitHoc web site [1].

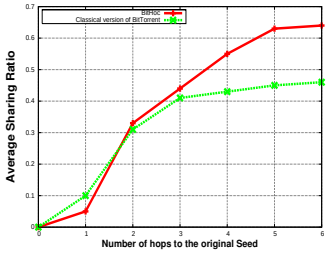

Figure 4: Sharing Ratio

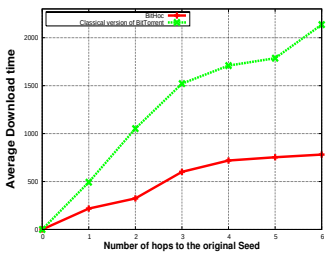

Figure 5: Download Time

5. REFERENCES

[1] BitHoc: http://planete.inria.fr/bithoc/.

[2] Bittorrent protocol: http://wiki.theory.org/bittorrentspecification.

[3] OLSR: http://www.grc.upv.es/software/intro_olsr.html.

[4] A.Krifa, M.K.Sbai, C.Barakat, and T.Turletti. BitHoc: A content sharing application for wireless ad-hoc networks, demo description in IEEE Percom, Texas.

[5] M.K.Sbai, C.Barakat, J.Choi, A. Hamra, and T.Turletti. Adapting BitTorrent to wireless ad-hoc networks, in Ad-Hoc Now, Sophia Antipolis.

[6] M.K.SBAI, E.Salhi, and C.Barakat. A membership management protocol for peer-to-peer services in MANET, INRIA technical report, http://hal.inria.fr/inria-00342691. 2009.

[7] M. P. and U.-K. G. Performance analysis of cooperative content distribution for wireless ad hoc networks, WONS, Obergurgl.