A Portal-based P2P System for Large Dataset Distribution

M.A. Baker and R. Lakhoo
School of Systems Engineering
University of Reading
r.n.lakhoo@rdg.ac.uk

ABSTRACT
In this paper, we present our initial work on a Portal-based peer-to-peer (P2P) system for large dataset distribution. Our aim is to create infrastructure that can be used by the scientific community, on the basis of other social networks, help download and maintain large datasets. We first outline some of the difficulties handling very large scientific datasets. Then we discuss P2P file sharing and storage applications that are commonly used today. We then outline our hierarchical P2P system design, which is based around a shared-portal service and our unique mini-peer that is embedded in a Web browser. To verify our system we have been investigating P2P simulation packages that will be used to emulate various set up and utilisation scenarios. In the near future work, based on initial investigations we will implement our system and test it by managing very large datasets.

Author Keywords
Peer-to-Peer, portal technologies, Content Distribution Networks.

INTRODUCTION
Peer-to-Peer technologies are commonly used to distribute content as part of a Content Distribution Network. One of the main reasons for the success of P2P networks is providing users with the ability to distribute content by contributing bandwidth to network. This concept is also related to the social aspect of a P2P network. Most, if not all, members of a P2P network contribute resources to the network. In recent years, applications such as BitTorrent (BT) [1] have become popular; it is a well-known method of distributing content too. Typically, BT is used to distribute GNU/Linux OS distributions and other multimedia files. However, the typical file size of content using BT is in the Mbytes or Gbytes range, while scientific communities that use the likes of the Sloan Digital Sky Survey (SDSS) [2], typically have datasets of the order of many Tbytes – and increasing. The SDSS project is set to generate approximately 15 Tbytes of data, during the project's five-year lifetime [3]. Obviously, the storage and distribution of such dataset is costly in terms of time and bandwidth. Mechanisms currently in place for the distribution of such a dataset come in two options. Firstly, to physically ship hard disks to a source that is willing to create a copy of the database, or undertake a point-to-point transfer from an institute, which hosts a copy of the data, using scripts or `wget`.

We propose a P2P system for scientific communities, which can combine the storage and bandwidth from peer contributions, to serve datasets that would not be normally possible or feasible to be hosted by individual project partners, or other interested parties. Our P2P system has a two-level hierarchy that uses a portal-based service as a backbone infrastructure and a Web browser component as mini-peers. The aim of our system is to create infrastructure that can be used by the scientific on the basis of social networks to help each other download and maintain large datasets.

BACKGROUND
There are many P2P systems currently available for file downloading. We describe the most popular ones here. Freenet [4] is an anonymous P2P file storage and retrieval system. Its primary aim is to allow users to publish and obtain information on the Internet without fear of censorship. The network is decentralised and its users are anonymous. Anonymity is provided by encrypted connections that are routed through other nodes on the network, making it difficult to find the source of a request. Users contribute bandwidth and storage space on their hard disk, but do not have control over what is stored on their computers. Freenet manages files by only keeping content that are popular, due to the finite amount of space available [5].

BitTorrent is a popular P2P file sharing system, which has forced ISPs to change policies and charging schemes [6]. BitTorrents success is mainly due to its distributed structure and P2P algorithms. It splits files into smaller pieces, which are known as chunks. Content within the network is published in a text-based `.torrent` file., which contains metadata about a file in the network, such as name, length, chunk hashing information and the location of the tracker. Trackers use HTTP to communicate with peers and help with the discovery of other peers in the network who are downloading the same content. Peers use the `.torrent` file with a BitTorrent client to download and upload chunks of a file to other interested peers. BitTorrents algorithms ensure fairness within the network. The primary algorithm is Tit-For-Tat (TFT), which gives a degree of certainty that users contribute bandwidth to other users. Other algorithms and selection processes are involved with the BitTorrent protocol, which includes chunk selection, choking algorithms and anti-snubbing [7]. Because BitTorrent distributes the task of searching for `.torrent` files and the discovery of peers to other resources, the vast majority of a user's bandwidth is efficiently used for the distribution of content.

A more extensive survey of P2P file storage systems [8] discusses other such efforts, such as the Cooperative File Storage (CFS), PAST, Ivy, OceanStore, and Farsite. None of the systems, bar Freenet, have been deployed into the 'real-world' and the bandwidth provided by these systems is insufficient to be used with data intensive applications.

SYSTEM DESGIN

General Overview

Our P2P system consists of a two-layered peer hierarchy, see Figure 1 for an overview. The peers are arranged in a hierarchy of portal, and volunteer peers. The backbone tier (portal peers) consists of a portal service hosted, potentially by, academic institutes or research laboratories, which are part of a particular project. Not only is the backbone able to contribute greater resources, in terms of storage and bandwidth, but it is also connected socially to the content they wish to distribute and store. A portal provides each hosting partner, with a collaborative application interface to the content, based upon standardised portlet technologies. Portlets installed within the portal will provide a community with Web-based tools to update, manage and monitor the dataset being distributed. A pluggable utility will be run at the originating source of the dataset. The utility will provide the initial bootstrapping of the content to the portal peers. The bootstrapping process is the distribution of the data to the portal peers and adds metadata about the dataset/parts. The portal peers will host at least a single copy of the content, which will be distributed to the other peers.

The volunteer peers consist of individuals and/or organisations that are either involved or interested in a particular project or dataset. Volunteers will run a mini-peer within their Web browser that contributes to the distribution of content. The mini-peer will use technologies such as Ajax and JavaScript to facilitate the chunk transfer. There will be many mini-peers per a portal peer. All peers donate an unknown amount of resources to the network for an unknown amount of time. Volunteer peer are expected to be general users (home, academic, portal and Web surfers). The P2P system uses volunteers to provide additional bandwidth and storage, for a hosting community. In addition, the extra replication of the content provided by the volunteer peers, will facilitate lower costs for the community interested in a dataset.

Each portal, will host part of a dataset, which is stored as chunks. The volunteer peers are the first set of peers to replicate the dataset from the portals to aid the distribution of the content. Each chunk also has an associated MD5 hash, to aid in the detection of corrupt chunks. Peers use the BitTorrent concept of a swarm to download chunks to and from each other.

Our P2P network will be based around the popular BitTorrent protocol, due to its efficiency in distributing content. The peers will provide a distributed service for the network, so that peers can find each other and chunks of data to download, this is commonly known as a tracker. This will be implemented using Tycho [9], a virtual registry and messaging framework. The distributed tracker will provide a central virtual point for the discovery of peers and file chunks, while providing resilience to a single point of failure. Each peer will use the virtual registry to get information about the location of available chunks. In addition, volunteer peers will use the distributed tracker to find, which chunks are to be replicated to aid peers.

The volunteer peers only have a finite amount of storage space for the replication of chunks. Therefore, the volunteer peers must replace chunks when they have no more storage space for new chunks. For these peers to be effective they must store chunks that will help other peers in the network. The chunks served from the volunteer peers must be the most needed chunks. This will ensure that peers can efficiently use the available bandwidth from the volunteer peers.

Tycho will be integrated into the portal as a service interfaced via a management portlet that is available to the users of a portal. This will provide the system with wide-area communications, so that state of the services installed in different portals are synchronised. This provides each hosting portal with a global view of the dataset being distributed. Moreover, it provides the ability for different members of different portals to collectively manage the data. Portals provide a security framework and access-based control for its tools and services. Project members can also use the tools to develop their own custom portlets to analyse or manipulate the data. This allows for collaboration between different members of a project.

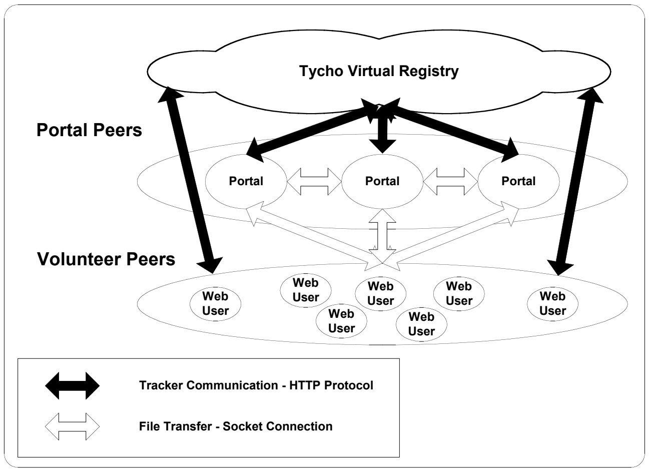

Figure 1, Portal-based P2P Network Overview.

The vision for the Portal-based P2P Network is for institutes working with large datasets to utilise the system across an institute, department, or project to support their research. Institutes cannot only distribute and store their data with our system, but also interact, socially, using the portal. It is hoped that the social bindings provided by the portals can also be used to distribute results, test data and updates to interested parties.

P2P Network Simulations

Implementing a distributed system is a more difficult task in comparison to traditional network programming, especially where fault diagnosis and failure are concerned. P2P systems add further complications, especially when testing different topologies and set–up scenarios across multiple machines. Finding stable wide-area network connections and suitable machines on these links is also a problem and can be impractical. This is because such an environment is not fully controllable, and results gathered may not be consistent. The time taken to develop such a system can be extensive, due to these and other issues. Thus, it was decided that a P2P simulator should be used, to model our network, experiment with different set-ups and configuration scenarios. This would allow us to, in theory, to test and develop our solution off-line and potentially with greater assurance.

A P2P simulator can allow us to model our system in particular scenarios, which will help us fine-tune our model. From the simulator results, an iterative approach will be used to update our model. Scenarios we are currently modelling includes situations where there are only a few volunteer peers, each providing a fixed amount of bandwidth. This will help us understand the optimised of volunteers needed to make a substantial contribution to the network. Scenarios that test the affect of fluctuations in number of volunteers will provide evidence for an acceptable minimum connection time. The joining and leaving from a P2P network is known as the flash-crowd effect and has been highlighted [10] as being an important aspect of a network. Related to this scenario is what size chunks should a volunteer distribute, to give optimum bandwidth to the lower tier. The chunk size used in P2P networks has been seen to have an affect on the distribution rate [11]. This scenario will also be used to ensure that volunteer resources are not depleted, and provide fairness within the network. The simulations should also help us identify optimum caching strategies, as well as memory and disk use by the peers. These scenarios aim to provide information on bottlenecks in the system and possible solutions.

We are currently investigating suitable simulators for our P2P system. The simulator needs to be able to implement our protocols, topologies and conditions for our configurations. Other scenarios are being developed, but as an initial stage, these few have been devised. Our aim is to use a suitable simulator to emulate and partially implement our P2P system. It is planned that the results from the simulations will provide an improved solution for implementation and testing outside of a simulator.

We have investigated, installed and tested several simulators and some are more suitable to our needs than others. The simulators are required to:

- Have support to implement custom P2P protocols,
- Provide facilities for multi-tier topologies,
- Provide visualisations,
- Provide reasonably accurate results in terms of 'real-world' performance,
- Have good support and/or documentation,
- Preferably, interface with the Java.

Our short-list consists of three simulators, general purpose P2P simulators, (GPS) [12], AgentJ [13] and OverSim/OMNET++ [14]. None of the simulators we have looked at completely fulfils our requirements, however, the three simulators are the closest and most feasible matches. GPS implements the BitTorrent protocol, is written in Java. This simulator is termed as being a general P2P simulator and is designed to allow users to develop other protocols. Stability issues have been noted with GPS, when a large number of nodes are used, these issues are currently under investigation. It has also become apparent that the GPS simulator is tightly coupled to the BitTorrent protocol, making it difficult to implement our own protocol. This issue has also been seen before [15] when trying to implement the Chord protocol. Its creator provides GPS support and documentation is currently being written. AgentJ provides developers with a means of deploying Java applications into the simulator, with minor changes to the source code. It uses the popular and well-established NS-2 simulator. An advantage of this simulator is that code to execute the simulation is a normal Java application, which can be used in the 'real-world' with minor changes. Although this may be useful, the time taken to develop a suitable application, does not merit many advantages over the implementation and testing on a suitable test bed. In addition, AgentJ seems to lack visualisation, thus, other software such as NAM [16] may be required to produce a visual representation of the network. OverSim is an overlay network simulation framework for OMNET++ and achieves the simulator requirements. Although, OMNET++ allows the use of Java to create modules with the JSimpleModule extension, it is not clear what the limitations are. The documentation states that a full protocol cannot be implemented with just Java and certain coding conventions must be applied, as the JSimpleModule is a wrapper for the underlying C++. The installation process for OverSim/OMNET++ is lengthy and a patch was required from the OverSim mailing list to make it operational. Currently OverSim implements Chord, GIA and Kademila overlay network protocols.

FUTURE WORK

The next stage of our work is to implement our P2P system in a simulator. We will execute our scenarios in the simulator and analyse the results. Using an iterative approach, we will optimise and update our model, then create further scenarios. This will provide us with a tested topology and a base design for further protocol development. The simulation alone is not a definitive conclusion on the performance or design of the P2P system. We will compare our simulation results with real implementations, which will be emulated on a test bed. The results from the test bed will be feed back into our development process. This should show the differences and unaccounted anomalies between the simulation and emulation. In addition, the performance of the two experiments can also be analysed. The next step is to deploy our implementation on machines hosted at different institutes. Although our test bed will run a real implementation, it does not take into account other Internet traffic, anomalies and other network restrictions, such as firewalls. This will start to our systems 'real-world' scenarios. An option at this point would be to use a system such as PlanetLab, which can potentially provide us with a large set of nodes distributed over a wide-area, to create a larger volume of nodes and traffic.

CONCLUSION

Progress is being made at a rapid rate with our Portal-based P2P Network. Our aim is to develop the system and make it available for other projects to use in the 'real-world'. This paper has presented initial work on a Portal-based P2P Network, designed to distribute and share large datasets. Our design is unique in combining the social aspect of a large dataset transfer with the workings of a P2P network, along with our Web browser-based mini-peers. The portlet-based tools allow the management of the dataset have also been discussed. We have also reviewed similar systems and presented our design for the P2P network. A number of P2P simulators have been reviewed, with three candidates currently being assessed further for use in our research. We have also presented scenarios for the P2P simulations and discussed future work in this area. It is hoped that the Portal-based P2P Network will find itself a useful contribution to projects with large and difficult to manage datasets, such as SDSS.

REFERENCES

[1] BitTorrent, http://www.bittorrent.com/.

[2] Sloan Digital Sky Survey, http://www.sdss.org.

[3] SDSS Storage Server Technical Note, http://home.fnal.gov/~yocum/storageServerTechnicalNote.html.

[4] Freenet, http://freenetproject.org.

[5] Protecting Freedom of Information Online with Freenet, http://freenetproject.org/papers/freenet-ieee.pdf.

[6] BBC News. BitTorrent battles over bandwidth, http://news.bbc.co.uk/1/hi/programmes/click_online/4905660.stm.

[7] Cohen. B. Incentives Build Robustness in BitTorrent, (2003). http://www.bittorrent.org/bittorrentecon.pdf.

[8] Hassan. R., Anwar. Z., Yurcik. W., Brumbaugh. L., and Campbell. R., A Survey of Peer-to-Peer Storage Techniques for Distributed File Systems. *Proc ITCC'05*, ACM (2005), 205-213.

[9] Baker. M.A., and Grove. M., Tycho: A Wide-area Messaging Framework with an Integrated Virtual Registry, Special issue of the Journal of Supercomputing, G.A.Gravvanis, J.P. Morrison and G.C. Fox, Springer US. (2006), ISSN 1573-0484.

[10] Guo. L., Chen. S., Xiao. Z., Tan. E., Ding. X., and Zhang. X., Measurements, Analysis and Modeling of BitTorrent-like Systems. Proc IMC'05. ACM (2005). 35-48.

[11] Yang. X., and Veciana. G., Performance of peer-to-peer networks: Service capacity and role of resource sharing policies, Performance Evaluation. Volume 63, Issue 3, P2P Computing Systems (2006), 175-194.

[12] Yang. W., and Abu-Ghazaleh. N., GPS: a general peer-to-peer simulator and its use for modeling BitTorrent. Proc MASCOTS'05. IEEE (2005), 425-432.

[13] AgentJ. http://cs.itd.nrl.navy.mil/work/agentj/index.php.

[14] OverSim: The Overlay Simulation Framework. http://www.oversim.org/.

[15] Naicken. S., Basu. A., Livingston. B., and Rodhetbhai. S., A Survey of Peer-to-Peer Network Simulators, Proc The Seventh Annual Postgraduate Symposium, Liverpool, UK, (2006).

[16] NAM: Network Animator. http://www.isi.edu/nsnam/nam/.