Towards Self-managing QoS-Enabled Peer-to-Peer Systems

Vana Kalogeraki, Fang Chen, Thomas Repantis, and Demetris Zeinalipour-Yazti

Department of Computer Science and Engineering, University of California, Riverside, CA 92521
{vana, fchen, trep, csyiazti}@cs.ucr.edu

Abstract. Peer-to-peer systems that dynamically interact, collaborate and share resources are increasingly being deployed in wide-area environments. The inherent ad-hoc nature of these systems makes it difficult to meet the Quality of Service (QoS) requirements of the distributed applications, thus having a direct impact on their scalability, efficiency and performance. In this paper we propose adaptive algorithms to meet applications QoS demands and balance the load across multiple peers. These comprise (a) resource management mechanisms to monitor resource loads and application latencies and (b) self-organization algorithms to dynamically select peers that maximize the probability of meeting the applications’ soft real-time and QoS requirements. Our algorithms use only local knowledge and therefore scale well with respect to the size of the network and the number of executing applications.

1 Introduction

In the last few years, the new emerging Peer-to-Peer (P2P) model has become very attractive for developing large scale file systems [1,2,3,4,5] and sharing resources (i.e., CPU cycles, memory, storage space, network bandwidth) [7,8] over large scale geographical areas. This is achieved by constructing an overlay network of many nodes (peers) built on top of heterogeneous operating systems and networks. P2P systems present the evolution of the client-server model that was primarily used to manage small-scale distributed environments. The most distinct characteristic in the P2P overlays is that there is symmetric communication between the peers; each peer has both client and server role.

Many efforts have been made to improve resource usage, minimize network latencies and reduce the volume of unnecessary traffic incurred in large-scale P2P overlays. Two main approaches have emerged for constructing overlay networks: Structured and Unstructured overlays. Structured overlay networks [4,5] are organized in such a way that objects are located at specific nodes in the network and nodes maintain some state information, to enable efficient retrieval of the objects. These sacrifice atomicity by mapping objects to particular nodes and assume that all nodes are equal in terms of resources, which can lead to...
bottlenecks and hot-spots. In unstructured overlay networks, on the other hand, objects can be located at random nodes, and nodes are able to join the system at random times and depart without a priori notification. Recent efforts have shown that a self-organizing unstructured overlay protocol maintains an efficient and connected topology when the underlying network fails, performance changes, or nodes join and leave the network dynamically [9]. More advantages of unstructured overlay networks include their ability for self-organization, for adaptation to different loads, and for resiliency to node failures. Several efforts have demonstrated that P2P systems can be used efficiently in the context of multicast [10], distributed object-location [4,8] and information retrieval [11].

However, hosting distributed, real-time applications with Quality of Service (QoS) demands, such as predictable jitter and latency on P2P systems imposes many challenges. These types of applications have distinctly different characteristics from content-based or multicast applications traditional being deployed on P2P systems. Examples of such applications include industrial process control systems, avionics mission computing systems and mission-critical video processing systems [12].

For example, consider a surveillance system that transfers public health, laboratory, and clinical data over the Internet. In this example, both continuous and discrete data (such as text, images, audio and video streams and control information) needs to be collected from multiple nodes in the system. Personnel will then analyze the gathered data quickly and accurately to monitor disease trends, identify emerging infectious diseases or track potential bioterrorism attacks. These have end-to-end soft real-time and QoS requirements on data transmission, including fast and reliable transfer, and substantial throughput. In addition, the audio and video streams may need to be transcoded to different formats or presentations (such as lower resolution) to transmit the data over resource constrained links. To support the QoS demands of the distributed applications, the P2P system must be flexible, predictable and adaptable.

Distributed and real-time applications have been successfully developed over middleware technologies, such as OMG’s Common Object Request Broker Architecture (CORBA) [13], Microsoft’s Distributed Component Object Model (DCOM) [14], Sun’s Java Remote Method Invocation (RMI) [15] and the Simple Object Access Protocol (SOAP) [16]. These typically rely on local management or the use of centralized managers that have a global view of the system [17], [18], [19], [20], [21].

In our view, the inherent advantages of the P2P systems, including scalability, decentralization and ease of use makes it feasible to develop large-scale distributed and real-time applications. However, current P2P systems are limited in capability because of lack of automated and decentralized management mechanisms. There are two main reasons for this limitation: (1) in a large scale system, each node cannot have an accurate global view of the system at all times, since the state of the system changes much faster than it can be communicated to the peers, and (2) the P2P infrastructure can encompass resources with different processing and communication capabilities, therefore, distributed