Available online at www.sciencedirect.com
SCIENCE DIRECT®
The Journal of Systems and Software 79 (2006) 523–536
The Journal of Systems and Software
www.elsevier.com/locate/jss

Declarative programming of integrated peer-to-peer and Web based systems: the case of Prolog

Seng W. Loke *

School of Computer Science and Software Engineering, Monash University, 900 Dandenong Road, Caulfield East, Melbourne, Vic. 3145, Australia

Received 20 August 2003; received in revised form 6 April 2005; accepted 6 April 2005
Available online 10 May 2005

Abstract

Web and peer-to-peer systems have emerged as popular areas in distributed computing, and their integrated usage permits the benefits of both to be exploited. While much work in these areas have utilized the imperative programming paradigm, the need for declarative programming paradigms is increasingly being recognized, not only for the often cited advantages such as a higher level of abstraction and specialized language features, but also to tackle the querying and manipulation of knowledge and reasoning with semantics that will be the mainstay of the proposed next generation of the Web and peer-to-peer computing. This paper presents an approach towards integrative use of the Web and peer-to-peer systems within a declarative programming paradigm. We contend that logic programming can be useful in peer-to-peer computing, especially for querying and representing knowledge shared over peer networks, and for scripting applications that involve sophisticated search behaviour over peer networks. As an example of peer-to-peer querying expressed in a logic programming language, we propose a simple extension of Prolog, which we call *LogicPeer*, to enable goal evaluation over peers in a peer network. Using LogicPeer, we outline how a peer-to-peer version of a Yahoo-like system can be built and queried, and several other applications that involve decentralized knowledge sharing. We then show how LogicPeer can be used with *LogicWeb*, a Prolog extension to access Web pages, thereby integrating peer-to-peer querying and Web querying in a common declarative framework.

Keywords: Prolog; Integration; Logic programming; Web computing; Peer-to-peer computing; Declarative programming

1. Introduction

In recent years, *peer-to-peer computing* has been given tremendous attention by technologists, businesses, and trend watchers. Peers share or exchange computer resources and services by direct exchange, i.e. there is a decentralization away from heavy-weight servers to equal-weight peers. The CPU cycles, disk storage, and files on a computer can be utilized by other peers leading to distributed storage which integrates the available disk space of a number of computers, distributed parallel processing, and distributed content management. Such decentralization can provide greater fault-tolerance since there is no single point of failure, distributed resource control and maintenance (e.g., distributed content management where individuals maintain the content they publish and have a sense of control over the content), greater efficiency of information exchange since traffic is not routed through central bottlenecks, and ease of set-up and usage which partly comes from distributed resource administration and in situ information sharing (e.g., information need not be transferred to another host such as a central server to be published).

With content distributed among peers in a distributed network, mechanisms including protocols and query languages for performing searches over the distributed

* Tel.: +61 3 99032614; fax: +61 3 99032863.
E-mail address: swloke@csse.monash.edu.au

0164-1212/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2005.04.005

resources are an obvious need. The Gnutella protocol,[1] has been used as the basis of several peer-to-peer systems for searching distributed content over a network of peers using simple keyword queries. Sun's new project JXTA,[2] which provides a shell for peer-to-peer programming, comes with basic built-in commands and protocols for searching within a group of peers. Gnutella,[3] FreeNet,[4] and SETI@home,[5] are examples of systems which have shown how the peer-to-peer model can be used for sharing resources directly (e.g., MP3 file-sharing) with little or no intervention from a central server or authority.

Logic programming languages such as Prolog have been explored for Internet applications (Davison, 2002) resulting in high level models, and extensions for manipulating Web contents. Logic programming provides a higher level of abstraction than imperative languages for knowledge representation and processing. Also, logic programming languages such as Prolog has features such as backtracking search, pattern matching, declarative syntax, and meta-programming. For data representation, deductive databases generalizes relational databases.

In this paper, we apply the declarative programming paradigm for integrating Web and peer-to-peer computing by providing lightweight extensions of Prolog for this purpose. While we use Prolog, we see that the approach can be utilized in other logic programming languages and formalisms that are emerging such as Mercury (Somogyi et al., 1995) and other Semantic Web logic languages.[6] Prolog provides a useful basis to illustrate and develop our ideas due to its simple semantics, widespread familiarity, and more than twenty years of research to leverage on.

Our contributions in this paper are as follows. In Section 2, as an example of peer-to-peer queries expressed in a logic programming language, we describe a simple extension of Prolog with operators to access peer networks, which we call *LogicPeer*, to enable goal evaluation over peers in a peer network (which is not done by Nejdl et al., 2001). We argue that logic programming can be useful for querying and representing knowledge shared over peer networks, and for scripting applications that involve heuristics based sophisticated search behaviour over peer networks. We do not advocate logic programming as a replacement of current languages for all peer-to-peer applications but aim to highlight the benefits of logic programming for specific kinds of peer-to-peer applications. In Section 3, using this extended Prolog, we outline how a peer-to-peer version of a Yahoo-like system can be built and queried. We also briefly consider several other applications that involve decentralized knowledge sharing. Section 4 presents another extension to LogicPeer goals to enable cooperative goal evaluation. In Section 5, we show how our extended Prolog can be used with LogicWeb (Loke and Davison, 1998) thereby integrating peer-to-peer querying and Web querying in a common framework. We consider related work in Section 6 and conclude in Section 7.

We assume an acquaintance with Prolog in the discussions below.

2. Extending Prolog for peer-to-peer querying

2.1. Characteristics of peer-to-peer computing

We make the following assumptions about a peer network:

- Each peer has a unique identifier called the *peer identifier*. This can be an IP address and port number but need not be so. A peer-to-peer system normally has its own namespace. For example, ICQ[7] has ICQ numbers to identify peers which are independent of network addresses. We do not assume any particular format for peer identifiers but only that they exist. We also assume the existence of name mapping mechanisms (e.g., to resolve ICQ numbers to actual IP addresses). We show that LogicPeer is independent of the format of peer identifiers and the underlying mechanisms for resolution.
- Peers can send messages among each other using peer identifiers.
- A peer does not have the peer identifier of every other peer in the peer network but we assume that a peer has a set of peer identifiers for peers it knows about, which we call the *friends* of the peer. We assume that a peer has some (perhaps out-of-band) means of discovering other peers in the network in order to build or update this set of friend identifiers. Peer directories and IP multicast might be used for discovering friend identifiers as in Napster servers, Gnutella reflectors,[8] or JXTA peer directories.
- In our model, each peer has a collection of Prolog rules (and facts) against which goals it receives from other peers and goals initiated locally are evaluated. This collection of Prolog rules might include the knowledge base to be shared by this peer. This main-

[1] Gnutella protocol specification is at http://capnbry.dyndns.org/gnutella/protocol.php.
[2] http://www.jxta.org
[3] http://gnutella.wego.com/
[4] http://www.at-web.de/p2p/freenet.htm
[5] http://setiathome.ssl.berkeley.edu/
[6] See http://www.semanticweb.org/inference.html for a list.
[7] http://www.icq.com
[8] See http://www.clip2.com.