Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 31 (2008) 629–642
computer communications
www.elsevier.com/locate/comcom

Computing with data non-determinism: Wait time management for peer-to-peer systems

Javed I. Khan *, Asrar U. Haque

Media Communications and Networking Research Laboratory, Department of Math & Computer Science, Kent State University, 233 MSB, Kent, OH 44242, USA

Available online 15 September 2007

Abstract

One of the unusual challenges faced by peer-to-peer algorithms as opposed to classical distributed algorithms is that these have to compute with data non-determinism where there is no guarantee that data from a particular node will be delivered in time or delivered at all. Each distributed component not only has to work in such environment, but also has to decide how long to wait for peers to respond. A large wait can inordinately delay entire computation. A smaller wait can reduce the coverage of the algorithm and drastically degenerate the quality of final result produced. In this paper, we investigate a set of wait management schemas which tries to minimize the overall completion time and maximize the quality of solution. We present detail analyses of the schemas for various scenarios including large power law networks and forms of non-responsiveness. The work promises dramatic improvement in peer-to-peer search and other computations.

Keywords: P2P; Timeout schema; Distributed algorithm

1. Introduction

P2P has emerged as an important networking paradigm in recent years. The fabric of peer-to-peer systems is founded on various distributed algorithms such as searching, packet routing, scheduling, shortest path, k-best path, network connectivity, spanning tree, etc. computed over various overlay networks.

An interesting question if it is distinct from classical parallel and distributed computing systems (PDCS) paradigm? It seems one of the distinction lies in the treatment of unreliability of distributed nodes. Unlike classical distributed systems nodes and client servers programs built on them, it is primarily based on peers who can voluntarily join and leave the network. Peers can volunteer or deny letting others use its resources. Thus the distributed algorithms face scenarios not common in classical network such as dead-beat node, unreliable or busy peer, missing messages, authentication failure, selective cooperation/non-cooperation etc. We call such a computing environment made out of the distributed moody and autonomous entity as *moody and autonomous environment* (MAE).

It seems MAE offers a new challenge that is neither solved by the data-deterministic classical distributed systems or by the flavor of reliability assured by classical communication primitives (such as TCP, or message passing). It is true that some primitives such as TCP make an extra effort to recover the lost messages if possible. Thus, classical distributed systems have safely assumed perfect data arrival in their design and corresponding performance characterization. There now clearly seems to be a gap. Peer-to-peer systems nodes have to abandon the notion of perfect data and thus perfect computing. We call it data non-determinism. The situation is unavoidable especially when the system consists of thousands or millions of autonomous entities – such as peer-to-peer systems or real life

* Corresponding author.
E-mail addresses: javed@kent.edu (J.I. Khan), ahaque@kent.edu (A.U. Haque).

0140-3664/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2007.08.047

social systems. It gives rise to some interesting new problems. What if even after the recovery effort the communication returns unsuccessfully? Currently, each individual algorithm needs to be separately programmed how an entity in MAE would proceed if the network (such as TCP or UDP) does not return the value for which it has been waiting for. A data-deterministic algorithm can be quite efficient in a classical distributed system environment but it can have drastically reduced performance (or even may not work at all) – when faced with a MAE.

The distributed component in MAE needs to decide some new issues such as how long it should wait for data in a particular step? If it waits longer whether the solution provided by the algorithm will improve or not? It has to address how the quality of solution would be affected if some peers use data offered by some other peers received from previous exploration of the network. Painstakingly, each of the distributed algorithms needs to be custom built for the solutions to moodiness'. In this paper, we study several strategies for wait management in a peer-to-peer system so that distributed algorithms (such as search or any other computation) can effectively operate on MAE environment in non-data deterministic situation.

Before we present our schemas, we provide a brief glimpse into related works. There are few interesting works on advanced timer in multi-way communication. For example, Network Weather Service [1] – a Grid performance data management and forecasting service, is used to forecast timeouts. A RTO (retransmission timeout) selection has been proposed in [2] for multimedia to achieve the optimal trade-off between the error probability and the rate cost in order to maximize throughput. Timeout strategy has been proposed associating costs for waiting time and retransmission attempts [3] where the optimal timeout value is calculated by minimizing the overall expected cost. DTRM (Deterministic Timeouts for Reliable Multicast) is used to avoid negative feedback explosion [4]. DTRM ensures that retransmission caused by only one NACK from a receiver belonging to a sub-tree arrives early enough so that the timers do not expire in the other receivers in that sub-tree. Each node in a peer-to-peer system participates in a form of programmable multicast. In active network literature such concept of aggregation has been studied in [17] to merge collected feedback in multicast application called concast. Wolf and Choi [18] proposed an aggregation algorithm similar to concast with the provision of detecting packet loss and avoidance of indefinite waiting. However, anyone is yet to address the problem of multi-stage multi-way wait optimization. In a previous work [21], we addressed a non-equal timer scheme for dynamic polling in active routers.

Some of the schemas in this paper assume that the nodes have knowledge of the link delay. There has been relatively more works [5–9] on statistical estimation of network tomography. Techniques has been shown to closely estimates network-internal characteristics including delay distribution, loss, and delay variance by correlating end-to-end measurements for multicast tree. Earlier network tomography research focuses on multicast routing only whereas the bulk of the traffic is uni-cast. This has been recently complemented by estimating delay distribution by employing uni-cast, end-to-end measurement of back to back packets [10–12]. Comparison has been done between direct (using ICMP timestamp) and indirect (based on end-to-end measurements) estimates [13]. An algebraic approach to determine link delay between sites where tracers cannot be placed has been given in [14]. The tracers [14] collect data from a network by using *ping* and *traceroute*. "Cing" [15] has been proposed to measure network-internal delays by using ICMP timestamp. Overall, it seems this internet delay estimation is a relatively matured area and few commercial services are also now available.

Instead of presenting our result with respect to any specific peer-to-peer system, we formalize the analyses based on a *reference model* of peer-to-peer systems. We have simulated the reference-model running on large power law networks. We provide analysis showing the performance of various timer management schemas for non-data deterministic communication that can be employed to control the *quality of result* vs. *completion time* in the phases of peer-to-peer fabric algorithms. The paper is arranged as follows. We introduce the reference model in Section 2. We briefly discuss how the non-data deterministic fabric algorithm can be mapped to the reference model in Section 3. The studied timer schemas are discussed in Sections 4 and 5. Finally, their performance simulation results are provided in Sections 6 and 7.

2. Reference model

The reference model has *nodes*, *sessions*, and a session specific *overlay network*. It also has a set of message primitives and program components. Each peer in this *reference model* is called a node. Algorithms in a peer-to-peer group communication work via sessions. Any node may initiate a session. Other nodes propagate and respond through a set of formal *group messaging schemas* that propagates via the peer's overlay network. A session consists of one or more phases. A peer may take one of the three roles during a phase: *session initiator*, *synthesizer*, or *terminal*. The initiator starts the phase by sending the first message of a session. A synthesizer processes and propagates the messages. A *terminal* is a leaf node. Upon receiving a message, it returns a synthesized message and initiates the return trip of information, which also flows back towards the initiator while each synthesizer peer locally processes (synthesizes) and then propagates the information upstream. The general computability of a peer-particularly the ability to process messages in their own way is modeled by a set of six-program components (to be described shortly).

To model various patterns of peer communication the *reference model* identifies three types of messages: *request*,