DELIS

Dynamically Evolving, Large-Scale Information Systems

Towards Cooperative, Self-Organised Replica Management

Overview

David Hales, Andrea Marcozzi - University of Bologna - Dept. of Computer Science
Giovanni Cortese - University of Rome - Laboratori di Radiocoms (RadioLabs)

Abstract

Nodes in a network often store and serve content to other nodes. However, each has a finite capacity and if requests for specific content exceed the capacity of the nodes serving it then queries fail. It is generally not possible for a priori predictions of load demand. At given times some content may suddenly become popular and at others hardly requested at all. Hence over a given time period a population of nodes has a certain total capacity to serve requests (the sum of all individual node capacities) and some demand load (queries going to the nodes). Assuming nodes may replicate content from other nodes and delegate or redirect queries when their capacity is exceeded we present a simple node level protocol that leads to efficient outcomes by dynamically adjusting, replicating and redirecting. The approach is scalable, robust and resistant to certain kinds of free-riding.

The CacheWorld Scenario

We assume a population of *N* nodes which form a P2P overlay network. In addition to being part of the overlay, each node functions as a server responding to requests (queries) which come from clients outside of the overlay network (clients). The overlay links servers bidirectionally if they mutually replicate content. Two linked servers hold a copy of the others' site. We also assume servers have access to three services: a replication service that copies items between servers; a peer sampling service that supplies a random server from the overlay; a content server that serves or redirect queries as required. Figure 1 shows a schematic diagram.

Query Handling

Over a given time period nodes receive queries (load) from clients to serve their content item. Each node has a capacity, *C*, specifying the total queries it can serve in the given time period. If the load exceeds capacity then the node is said o be "overloaded". Overloaded nodes redirect queries to randomly selected neighbors.

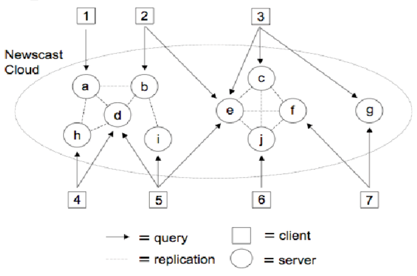

Fig. 1: Schematic of the CacheWorld Protocol

```
Passive thread
on receiving a query q, node i:
        if not overloaded, service q directly
        else if neighbors > 0 and q is not
        already a redirected query
                j ← selectRandomNeighbor()
                redirect q to j
        end if
```

```
Active thread
periodically each node i:
        if not satisfied
                drop all neighbor links
                j ← selectRandomPeer()
                if j is receptive then link to j
        end if
```

Fig. 2: Outline pseudocode for the protocol. The passive thread is activated when a node receives a query either directly or redirected. The active theread is activated periodically appoximately once per load cycle. The value *Ci* represents the capacity of node *i*. This is the maximum number of queries it can serve per load cycle.

If a neighbor is not itself overloaded it will serve the query from its local content replica, otherwise it will ignore the query.

Satisfaction and Movement

Each node maintains an estimate of the proportion of queries for its own content that are actually served (*ps*). A node is said to be satisfied when $ps \geq t$ where t is some threshold value. Periodically nodes attempt to change their neighbor (move) in the network if they are not satisfied. Figure 2 gives outline pseudocode for the protocol.

Experimental configuration and results

Our initial experiments have been performed with a small number of nodes *N = 50* bur we found our protocol to be broadly scalable. In our experiments all the nodes have a capacity of 20 (*C = 20*); as for the load (query received over a given time period) 10% of nodes had load 5, 10% of nodes had load 35, 40% of nodes had load 15 and the remaining 40% of nodes had load 25. This means that half of the nodes are overloaded and half of the nodes are underloaded. We set *k=4* meaning that each node has a maximun of 4 links to other nodes. We run 10 independent run and then took the average. Results are shown in figs 3 - 4.

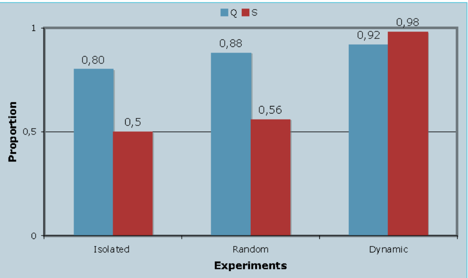

Fig. 3: Results from three experiments: *Isolated* gives a baseline where all nodes are isolated. *Random* gives a second baseline for a fixed random network. In *Dynamic* the CacheWorld protocol is fully active. Q = proportion of queries answered, S = proportion of nodes satisfied.

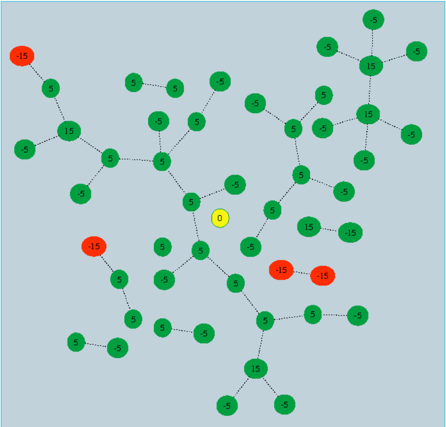

Fig. 4: A plot of the network after 1000 cycles. Number inside nodes indicates the needed / spare capacity. Red nodes are unsatisfied node. Grenn nodes are satisfied. The node in the center (the yellow one) is a "free rider" which results isolated: it always passes on its queries when linked, it uses its own capacity when isolated.

Related Work

Several Replica Management systems already exist and are in use. However existing systems which manage replication (*Guillaime et al.*), do not have specific mechanisms for adjusting the cooperation among nodes. For example, for balancing global policies (i.e. the goals of the community) against local (i.e. that of individual nodes / agents) policies. The policies in such systems are fixed by design, after using simulation to understand good trade-offs in the protocol design phases. Our approach does not represent any global policy but rather emerges that policy based on purely individual goals and actions. Each individual node, based on its own satisfaction evaluation decides if to find new replication neighbors.

Open Issues and Future Work

In our current experiments the load profile is fixed not dynamic. Additionally, we have currently not taken into account the cost or quality of the links between clients requesting content and servers. We have assumed this is constant. The current model is, for the sake of simplicity, ignoring the cost of replication, which is probably the highest priority next step in our research. Also, it is not considering the rate of "obsolesce" or updates of a content items.

References

D. Hales and S. Arteconi. SLACER: A self-organizing protocol for coordination in peer-to-peer networks. *IEEE Intelligent Systems*, 21(2):29-35, Mar/Apr 2006.

M. Jelasity, M. van Steen. Large-Scale Newscast Computing on the Internet. Report IR-503, Vrije Universitet Amsterdam, Dept. of Comp. Sci. 2002.

P. Guillaume and Maarten van Steen. Globule: a Collaborative Content Delivery Network. *IEEE Communications Magazine* 44(8), pp. 127-133, August 2006.

Contact 1
David Hales
E-Mail: hales@cs.unibo.it
URL: www.davidhales.com

Contact 2
Andrea Marcozzi
E-Mail: marcozzi@cs.unibo.it
URL: www.cs.unibo.it/~marcozzi

Dept. of Computer Science
Mura Anteo Zamboni 7
40127 Bologna, Italy
Fax: +39 051 2094510
Phone: +39 051 2094515

Contact 3
Giovanni Cortese
E-Mail: g.cortese@computer.org
Dipartimento di Ingegneria Elettronica
Via Arrigo Cavaglieri 26, 00173 Roma, Italy

DELIS: Integrated European Project founded by the "Complex Systems" Proactive Initiative within the Sixth Framework Programme