Building Low-Diameter P2P Networks

Prepared by : Sayak Mitra, Aruni Choudary, Monotosh Das

Introduction :

In a peer-to-peer (P2P) network, nodes connect into an existing network and participate in providing and availing of services. Unlike traditional client-server network, there is no central server. Each node runs both server and client using servent (server+client) software. On joining the network, this software locally decides which neighbors to connect to. In this paper a simple scheme for participants to build P2P networks in a distributed fashion is proposed, and it is prove that it results in connected networks of constant degree and logarithmic diameter. It does so with no global knowledge of all the nodes in the network.

Motivation :

Current P2P networks (e.g., Gnutella) are constructed by participants following their own un-coordinated (and often whimsical) protocols. They consequently suffer from frequent network overload and fragmentation into disconnected pieces separated by chokepoints with inadequate bandwidth. A new protocol has been proposed here for the network. The protocol results in a constant degree network that is likely to stay connected and have small diameter.

Case Study : Gnutella :

Gnutella is a public P2P network on the Internet, by which anyone can share, search for and retrieve files and content. Gnutella works as follows :

- At first the servent of the participant join the network by connecting to a small number of (typically 3-5) neighbors currently connected to the network.

- When any servent s wishes to search the network with some query q, it sends q to its neighbors.
- These neighbors return any of their own documents that match the query; they also propagate q to their neighbors, and so on.
- matching results are fanned back into s along the paths on which q flowed outwards.
- To control network traffic the fanning-out typically continues to some fixed radius (in Gnutella, typically 7).

The Proposed Network Protocol :

A new protocol is proposed by which newly arriving servents decide which network nodes to connect to, and existing servents decide when and how to replace lost connections. It is shown that this protocol results in a constant degree network that is likely to stay connected and have small diameter.

The central element of the protocol is a host server, which at all times maintains a cache of K nodes. The host server can be reached from all nodes. However, it need not know of the topology of the network or identities of the nodes. The protocol ensures that the degree (number of neighbors) of all nodes will be in the interval [D, C +1], for two constants D and C. A new node first contacts the host server, which gives it D random nodes from the current cache to connect to. After connecting to these nodes new node becomes a d-node. It remains a d-node until it subsequently either enters the cache or leaves the network. At some point the protocol may put a d-node into the cache. It stays in the cache until it acquires a total of C connections and then leaves the cache, as a c-node.

Steps Followed in the Algorithm :

1. On joining the network a node v connect to D cache nodes, chosen uniformly at random from the current cache.

2. If a neighbour of v leaves the network, and that connection was not a preferred connection, connect to a random node in cache with probability D/d(v), where d(v) is the degree of v before losing the neighbour.

3. When v reaches degree C in the cache, it is replaced in the cache by a d-node selected by Cache Replacement rule. The Cache Replacement rule works as follows :

   Let there are n nodes in the cache.
   0th node is the cache is the C-node to be replaced (v). Find the replacement node by:
   K = 0;
   while (a d-node is not found AND k<n)
   search the neighbors of k^{th} node of the cache for a d-node;
   k = k+1;
   endwhile

4. When v leaves the cache as a c-node it maintains a preferred connection to the d-node that replaced it in the cache. If v is not already connected to that node then create a new connection between these nodes.

5. If v is a c-node and its preferred connection is lost, then v reconnects to a random node in the cache and this becomes its new preferred connection.

The proposed network is a connected networks of constant degree and logarithmic diameter. Various other facts can be highlighted in the following analysis.

Analysis of the network :

- Let G(V,E) denote the network at time T.

- Consider a quantity $N=\lambda/\mu$ where λ denotes the node-arrival rate and μ the node-leaving rate.
- 1) if $T=\Theta(N)$, then $V= \Theta(N)$
- 2) if $T/N\rightarrow\infty$ then $V=N\pm o(N)$
- If at time t N changed to N', then at $T \rightarrow\infty$ then $V=N\pm o(N)$

Available Node capacity :

- Let C>3D+1;; then at any time $t \geq a\log N$ (a>0) there are
- Assuming C>3D+1. At any time $t \geq c\log N$, with probability $1-O(\log^2 N/N)$ the algorithm finds a replacement d-node by examining only O(logN) nodes.

Connectivity :

- At all times each node is connected to a Cache node either directly or through some path in the network.
- Consider two Cache nodes at time $T \geq c\log N$. With probability $1-O(\log^2 N/N)$ these nodes are connected.
- There is a constant c such that at time $T \geq c\log N$ $P(G \text{ is connected}) \geq 1-O(\log^2 N/N)$.

The Diameter Bound :

- For any T s.t $T/N\rightarrow\infty$, with high P, the largest connected component of G has diameter O(logN). The Probability of connectivity is $1-O(\log^2 N/N)$.
- Assume that node v enters the network at $T/N\rightarrow\infty$. Then for a sufficiently large value of C, probability that v leaves the cache as a good node is at least P>1/2.

The need of Preferred Connections :

The preferred connection component in this protocol is essential because running the protocol without it leads to the formation of many small

disconnected components. Consider the network at T>N. there is a constant probability that there exists an isolated component. The expected number of these components is Ω(N) in the absence of preferred connections.

Conclusion :

We saw that the simple scheme proposed here can be used to build P2P networks in a distributed fashion, and it results in connected networks of constant degree and logarithmic diameter. This can be achieved in current P2P networks like Gnutella. There is a minuscule probability of catastrophic failure : for instance, in the cache replacement step, there is a very small probability that no replacement d-node is found. The solution can be to allow some nodes to exceed C+1 connection, or reject new connection. It can be seen that rules 1, 2, 4 and 5 can be easily implemented with constant overhead. Replacing a full cache node (rule 3) is constant on the average, and with high probability is at most logarithmic in the size of the network.

References :

Gopal Pandurangan, Prabhakar Raghavany and Eli Upfal. Building Low-Diameter P2P Networks.