Building Small Worlds in Unstructured P2P Networks using a Multi-agent Bayesian Inference Mechanism

Prithviraj Dasgupta
Computer Science Department
University of Nebraska, Omaha, NE 68182.
E-mail: pdasgupta@mail.unomaha.edu

ABSTRACT
Over the past few years, peer-to-peer (p2p) unstructured networks have emerged as an attractive paradigm for enabling online interactions between a large number of users in a decentralized manner. However, the decentralized nature of unstructured p2p networks makes load balancing a challenging problem. Specifically, the self-interested nature of users on the nodes of a p2p network and dynamic changes in network topology give rise to an unbalanced distribution of nodes across an unstructured p2p network. This results in network congestion and significant search latencies for all nodes. In this paper, we describe a small-world network model and a Bayesian inference mechanism within a multi-agent setting to address these issues. Simulation results for a file sharing p2p application show that our algorithm achieves an exponential reduction in number of messages exchanged and improves load-balancing across the network.

Categories & Subject Descriptors: I.2.11 Distributed Artificial Intelligence: Multi-agent systems.
Keywords: P2P networks, small world graphs, topology balancing.

1. INTRODUCTION
A p2p network is an overlay network between nodes interconnected by an underlying physical network and enables users located on its nodes to establish short-term connections with other users(nodes) to acquire and share resources with each other in a decentralized manner. In the absence of any centralized mechanism for controlling the topology of the network and for load balancing, p2p networks are characterized by an unbalanced topology with high clustering of nodes in some regions and relatively sparse node distribution in others. This results in network congestion, significant latencies for resource search requests, and a disparate distribution of load across the p2p network. In this paper, we address the issues related to unbalanced topology for unstructured p2p networks.

In an unstructured p2p network, a user(node) wishing to join the network selects one or more existing nodes within the network as its neighbors. Once it has joined the network, a node sends information it wishes to disseminate across the network (for e.g. resource search queries in a p2p file sharing application) to its neighbors. The neighbors then forward the query across the network using a flooding mechanism. The unbalanced network topology in an unstructured p2p network can be attributed to two features of the p2p node-join mechanism: (1) When a node joins a p2p network, the selection of neighbors is decided only by the node entering the network, while existing nodes in the network unconditionally accept the entering node as a neighbor. (2) A node can join a p2p network network at any arbitrary location and potentially generate a large amount of network traffic. This gives rise to sporadic traffic patterns leading to an uneven distribution of network load. In this paper, we extend the model of a small-world structure for p2p networks described in [2] using a Bayesian inference mechanism that enables each agent to respond to dynamic changes in the network’s topology and a node’s parameters. 1 Building a small-world graph within an unstructured network is considerably challenging because nodes can possess any arbitrary degree and can join at any arbitrary location in the network.

2. SMALL-WORLD MODEL FOR UNSTRUCTURED P2P NETWORKS
Small-world networks are a class of networks [3] that have the following properties: (i) a short characteristic path length that is typical of random graphs, and, (ii) a high clustering coefficient that is typical of well-ordered graphs. These properties prevent the formation of hubs in the network and make small-world networks easily navigable. A small-world graph is constructed from a non-small-world base graph in two phases. First, edges are added between certain nodes of the non-small-world base graph to get an augmented graph that satisfies the small-world properties. The edges that are added can be of two types: (i) Short-range link that connects two nodes that are separated by a distance \( \leq L_s \) in the base graph, and (ii) Long-range link that connects two nodes that are separated by a distance \( > L_s \) in the base graph. The distance \( L_s \) is called the short-range distance of the small-world graph. Simultaneously, the number of links that are added is controlled to maintain a fixed ratio \( \alpha \) between the number of short-range links and the total number

\[ \text{We have modeled our unstructured p2p network using the Gnutella 0.6 protocol [4].} \]
of links (short range+long range) added to each node in the augmented graph. \( \alpha \) is called the proximity factor. A suitable value of \( \alpha = 0.66 \) to ensure small-world properties has been reported in [5].

3. UTILITY-BASED NEIGHBOR SELECTION

The parameters used by our agent-based neighbor selection algorithm are the following:

- \( N \) Set of agents(nodes) where each agent is located on a node of the p2p network.
- \( N_i \) Set of neighbor agents(nodes) of agent \( i \in N \).
- \( R \) Set of resources stored by agents(nodes) of the p2p network.
- \( \tau \) Set of resource types for the different resources \( R \) in the p2p network.
- \( r_{j,t} \) Number of resources of type \( R \) that is \( \tau \) shared by agent \( j \).

For our model, we define the following parameters to enable a priori calculation of resource availability on a node:

1. **Resourcefulness**: The resourcefulness of agent \( j \) to agent \( i \) for resource type \( t \) is given by \( \rho_{i,j,t} = \sum_{v \in N_i} \delta(i,v) \). Intuitively, \( \rho_{i,j,t} \) denotes the relative contribution of agent \( j \) to the total number of resources of type \( t \) shared by different agents with agent \( i \).

2. **Connectedness**: Let \( N(i,h,\rho_{min}) \) denote the set of agents that are \( h \) hops away from agent \( i \) such that each agent \( j \in N(i,h,\rho_{min}) \) satisfies the criterion \( \rho_{i,j,t} > \rho_{min} \) for resource type \( t \). The connectedness of an agent \( i \) for resource type \( t \) is given by \( \chi(i,t,k) = \sum_{h=1}^{k} \sum_{v \in N_i} \delta(i,v) \) where \( \delta \) is a control parameter \( 2 \) and \( k \) is the connectedness lookahead parameter. The connectedness value of an agent \( i \) is a measure of the number of agents with a minimum resourcefulness \( \rho_{min} \) that are within a radius of \( k \) hops from the agent, and reflects the expected availability of resources among agents in the vicinity of the agent.

We assume that the probability of locating a resource of type \( t \) at an agent \( j \) by another agent \( i \) is approximated from agent \( j \)'s resourcefulness and connectedness parameters and is given by \( \pi_{i,j,t,k} = \rho_{i,j,t} \times \chi(i,t,k) \). For each resource type \( t \), the connectedness of an agent \( i \) for resource type \( t \) is given by \( \chi(i,t,k) = \sum_{h=1}^{k} \sum_{v \in N_i} \delta(i,v) \) where \( \delta \) is a control parameter \( 2 \) and \( k \) is the connectedness lookahead parameter. The connectedness value of an agent \( i \) is a measure of the number of agents with a minimum resourcefulness \( \rho_{min} \) that are within a radius of \( k \) hops from the agent, and reflects the expected availability of resources among agents in the vicinity of the agent.

Let \( v \) and \( c \) denote respectively the average value of a resource and the average search cost/hop to agent \( i \). The utility to agent \( i \) by accepting agent \( e \) as a neighbor for resource type \( t \) is given by:

\[
U_{i,e,t} = \text{Utility from accepting } e \text{ as a neighbor} = (v \pi_{i,e,t} - c) - (v \pi_{i,e} - c)(\Pi_{j=t}^{\min} (1 - \pi_{j,i}))
\]

or,

\[
U_{i,e,t} = (v \pi_{i,e,t} - c)(1 - \Pi_{j=t}^{\min} (1 - \pi_{j,i}))
\]

Fig. 1 shows the decision function used by an agent to admit neighbors while maintaining the small-world properties.

4. BAYESIAN UPDATE MECHANISM FOR ADDRESSING AGENT DEPARTURES

In this section, we describe a Bayesian inference mechanism that enables an agent to dynamically update another agent’s \( \rho \) and \( \chi \) values without explicitly exchanging messages with that agent. We have modeled dynamic updates arising from agent departures through the changes to the resourcefulness(\( \rho \)) and connectedness(\( \chi \)) parameters of agents in the vicinity of the departed agent. We postulate that as long as most of the neighbors of an agent \( j \) remain unchanged, the value of \( \chi_j \) can be estimated from the connectedness of agent \( j \)'s neighbors. Based on this assumption, we describe a Bayesian update based mechanism that enables an agent \( i \) to approximate the resourcefulness and connectedness parameters of an agent \( j \) dynamically without exchanging any additional messages with agent \( j \) or its neighbors.

Suppose an agent \( i \) wishing to determine the utility of accepting entrant agent \( e \) as a neighbor discovers that it has not recently updated the connectedness \( \chi_{i,j,k} \) of agent \( j \) which lies along the ping-path from agent \( e \) to agent \( i \). Let \( C : \chi \rightarrow wc/wc \) denote a function that maps the connectivity of an agent to a boolean variable that takes two values: well-connected(wc) and not well-connected(wc’). We assume that the connectedness parameter of agent \( j \) is influenced by agents lying within a radius of \( k \)'s hops from it. Agent \( i \) is then interested in calculating the probability:

\[
P(C(\chi_j) = \text{wc} \mid C(\chi_{i,j} = \text{wc})), C(\chi_{i,j} = \text{wc}'),..., C(\chi_{i,j} = \text{wc}'))
\]

for \( k = 1, k' \), where \( C(\chi_{i,j} = \text{wc}) \) denotes the connectedness of the \( l \)-th agent that is \( k \)'s hops away from agent \( j \).

The probability that agent \( n \), a \( k \)'s hop neighbor of agent \( j \) is also an \( m \)-hop \((m \leq n+1) \) neighbor of agent \( i \) is given by:

\[
P(h_{n,i} = m | h_{n,j} = k') = \Delta^{k'-m+1}
\]

where \( \Delta \) is the clustering

---

**Figure 1**: Algorithm used by agent \( i \) to respond to a ping from an entrant agent \( e \).
A suitable value for $\Delta [i]$ is given by: $\Delta = N_2 \frac{\sum_{j \in E_i} d_j}{N_2}$, where $N_2$ is the number of agents in the sub-network located within a radius of TTL hops of agent $j$ with degree $\geq 2$, $E_i$ is the number of edges in the sub-network and $d_j$ is the degree of agent $i$. We can then rewrite the connectedness of an agent $h'$ hops away from agent $j$ as:

$$\chi_{a(i,j,m)} = \sum_{m=1}^{k'+1} (P(h_{m,i} = m|h_{m,j} = h') \times \chi_{a(i,m)})$$

or,

$$\chi_{a(i,j,m)} = \sum_{m=1}^{k'+1} (\Delta^{h'-m+1} \times \chi_{a(i,m)})$$  \hspace{1cm} (2)

Using Bayes’ rule and Equation 2 we can rewrite Equation 1 as:

$$P(C(X_j) = wc \mid C(X_{a(1,j,h')}), C(X_{a(2,j,h')}),...) = \frac{P(C(X_{a(1,j,h')}), C(X_{a(2,j,h')}),..., C(X_j) = wc) \times P(C(X_j) = wc)}{P(C(X_{a(1,j,h')}), C(X_{a(2,j,h')}),..., C(X_j) = wc)}$$

$$= \frac{P(C(\Delta^{h'-m+1} \times \chi_{a(i,m)}) C(\Delta^{h'-m+1} \times \chi_{a(i,m)}) C(\Delta^{h'-m+1} \times \chi_{a(i,m)})...) \times P(C(X_j) = wc)}{P(C(\Delta^{h'-m+1} \times \chi_{a(i,m)}) C(\Delta^{h'-m+1} \times \chi_{a(i,m)}) C(\Delta^{h'-m+1} \times \chi_{a(i,m)})...) \times P(C(X_j) = wc)}$$

Each term in the above equation is available at agent $i$. Therefore, the connectedness parameter $\chi_j$ of agent $j$ can be approximated by agent $i$ from its knowledge of the connectedness parameter of agent $j$’s neighbors.

5. SIMULATION RESULTS

For our experimental results we consider a p2p network comprising a maximum of 1000 agents. The arrival rate of agents follows a Poisson distribution with a mean = 50. The maximum degree of an agent is drawn from $\{3, 6, 9\}$. The values of the different constants used for the simulation are $k = 2$, $\delta = 2$, $\gamma = 0.5$, $v = 10$, $c = 2$, $\alpha$ (proximity factor) = 0.66, and $E_s$ (short-range distance for small-world network) = 3. Parameters $d$ (average agent degree) and TTL (time-to-live for a message) are varied across simulation runs. The period (in minutes) for which an agent remains in the network is drawn from $U[5-30]$. Each simulation was run over a period of 15 mins. The dataset used for the resources on each agent is drawn from a set of 10,000 files. Resources can be of one of three possible resource types, and each agent has a preferred resource type selected at random. Resource search queries are generated at intervals of 10 seconds from an agent selected randomly from the currently active agents. The query string for a query is drawn randomly from the file names in the dataset. All results are averaged over 10 runs. The simulation results are reported in Figures 2 through 4.

6. REFERENCES