The Circle:

Some first steps towards decentralizing internet services

Paul Harrison

pfh@csse.monash.edu.au

Motivation

Many internet services are based on a client server model. This has several drawbacks

- The server is a single point of failure
- The server has to be very powerful to handle the load
- The server is generally owned by a single entity that can make arbitrary decisions, impose tolls, and is susceptible to legal attack

Can we eliminate the server, and provide decentralized services that don’t have these problems?

Fundamental problem of decentralization

- Resource discovery

Examples:

- Find a particular file
  eg “file: keyword britney”
- Find a particular person
  eg “person: pfh”
- Find a computer that provides a particular type of service
  eg “service: data caching”

Previous attempts

Gnutella

- Floods the whole network each time someone searches it
- Not good if there are millions of nodes in the network!

FastTrack (KaZaA / Grokster / Morpheus)

- Like gnutella, but not all nodes participate in search
- Kind of like optimizing bubblesort

There’s got to be a better way...

A different approach

- Use a decentralized hashtable
- Find resources by looking up a hash of a string describing the resouce
- Split the hashtable over all computers in the network

Don’t need to search the whole network any more, because we know where the information we want is.

Similar projects

- Chord
  `http://www.pdos.lcs.mit.edu/chord/`
- GISP
  `http://gisp.jxta.org/`

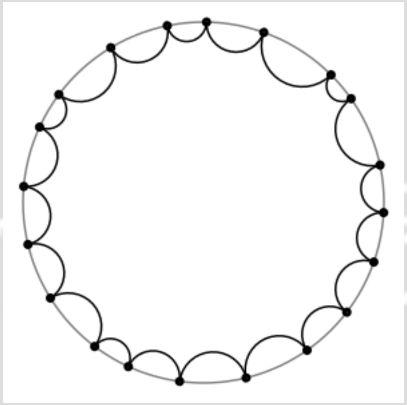
Design

- Map hashtable onto a circle
- Each node in the network chooses a random position on the circle
- Each node is responsible for the section of hashtable between its position and the position of its immediate clockwise neighbour

Design

Connectivity

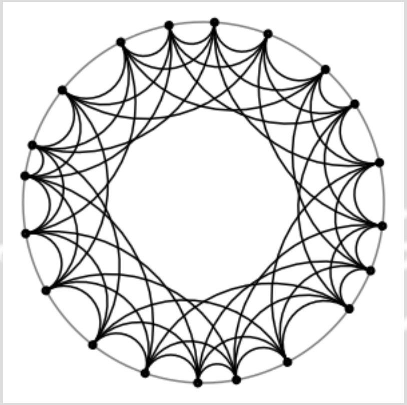
- Each node knows its immediate neighbours on the circle



Design

Connectivity

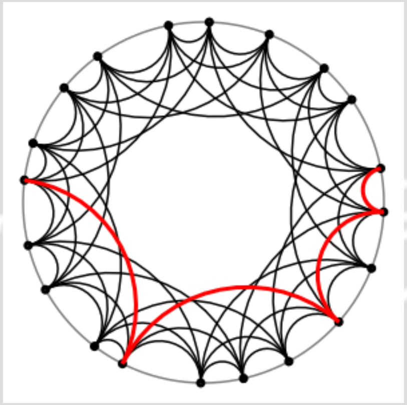
- Each node knows a few other nodes about the circle



Design

Connectivity

- Follow the links to find any other node in the circle quickly



Design

Advertising a resource

- Describe your resource with a string
  eg “file: keyword britney”
- Compute a hash of the string
- Follow the links around the circle until you find the node responsible for the hash
- Ask that node to store your advertisment

Finding a resource

- Follow the links to the appropriate node
- Ask that node for the location of nodes providing the resource you want

Design

Joining the network

- Choose a random position on the circle
- Locate at least one node on the network
- Follow links from the node you found to your position on the circle
- Say hi to your neighbours
- Fetch from your immediate counter-clockwise neighbour the section of hash table you have taken over

Design

But what if...

- ... one of your neighbours disconnects unexpectedly?
  Need to poll neighbours regularly
- ... a node storing information for you dies unexpectedly?
  Nodes storing information send a regular “I’m still here message”

For efficiency, poll at exponentially expanding intervals.

Implementation

Written in python

- Very experimental project
- Needed a language where I could quickly try out new ideas

All network communication is UDP based

- A search of the hashtable requires brief contact with several nodes
- No need to go to the trouble of setting up a TCP connection

Applications

File searching

- Search by keyword
- Extracts keywords from filenames and ID3 tags of MP3s
- When a node comes on-line it advertises each keyword of each file it has
- Supports downloading one file from multiple sources simultaneously

Applications

Instant messaging

- When you come on-line you
  1. Advertise that you are now online
  2. Advertise your first name, surname and username to allow people to search for you
  3. Advertise an interest in whether your various friends are online
  4. Inform people who are interested in you that you are online
- You can search for people, watch them come on and off-line, and of course, send messages
- Uses RSA signatures so that people can not pretend to be someone else

Applications

Not-so-instant messaging

- You can also send messages to people who aren’t currently online
- Your message can be stored to another node in the network so that even if you go offline the message can still be delivered
- Nodes storing a message advertise this fact so that the recipient can fetch the message when they come online
- When a node storing a message goes offline it passes the message to another node on the network
- Encrypted using RSA and Rijndael

Applications

Personalized news

- People nominate that they trust the news of other people to different degrees, forming a trust network
- The *trust distance* of a news item is the length of the shortest path through the trust network from the original source to yourself
- Nodes periodically query friends for interesting news
- Display news items in order of trust distance

Kind of like Advogato, but with you as the root, rather than raph et al

Future directions

Using the trust network for domain name resolution

- Any node can claim any name it wants
- When you want to resolve a name, you find the nearest node in the trust network that claims that name

This means

- Name allocation is very democratic:

  The person with the most trust gets the most out of a name
- Who owns a name depends on your perspective:

  An engineer types in “RMS” and finds a web page on measuring signal strength

  A programmer types in “RMS” and finds a web page on the ethics of information

Don’t yet have an algorithm to do this efficiently in a decentralized way

Future directions

An idea similar to the Circle can be used to do routing

- Each node chooses a random position on the circle
- Each node discovers physical routes to its immediate neighbours on the circle, and to some others about the circle

Then we can very easily find a route to any node in the network

- Follow the routes through the network, homing in on the node you want
- Concatenate all these routes to produce a route to that node
- Optimize the route using local optimizations

No need for an allocated IP address, no messy configuration, it just works

Conclusion

Scalable decentralized resource discovery is

- possible
- very useful

All of the centralized aspects of the internet can be replaced with decentralized substitutes. This would yield a network that

- makes good use of computing and network hardware
- deals gracefully with hardware failure
- is not under the control of any one organization
- has no entry tolls