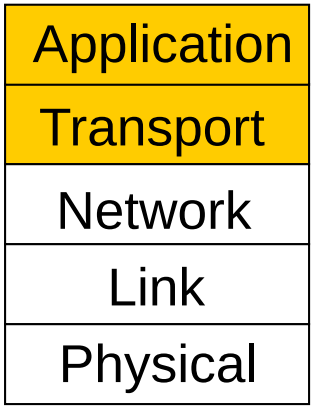
CSEP561 – Content Distribution

David Wetherall
djw@cs.washington.edu

Content Distribution

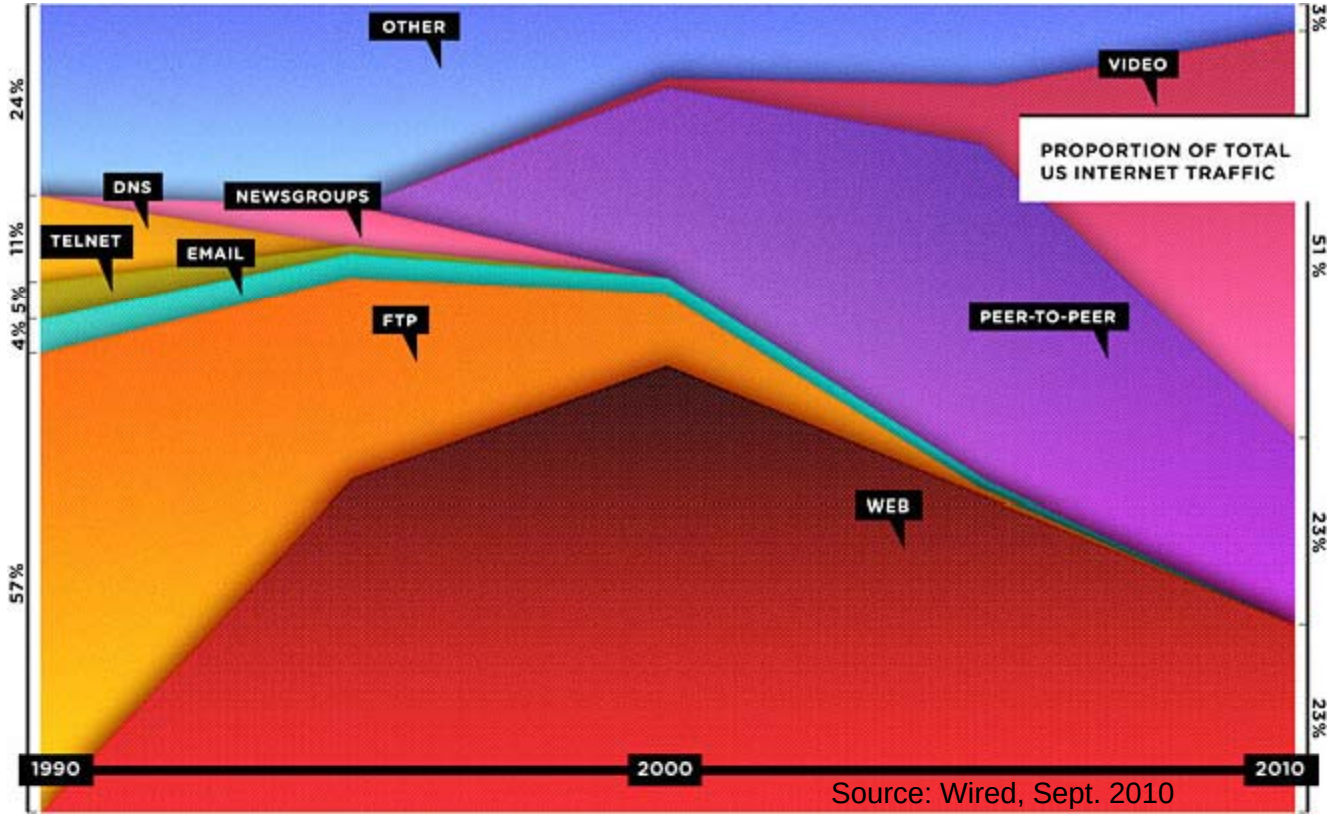
- Focus:
  - Things you should know about Internet workloads
  - Architectures for content distribution

- Traffic characteristics
- Caching
- CDNs
- Peer-to-peer



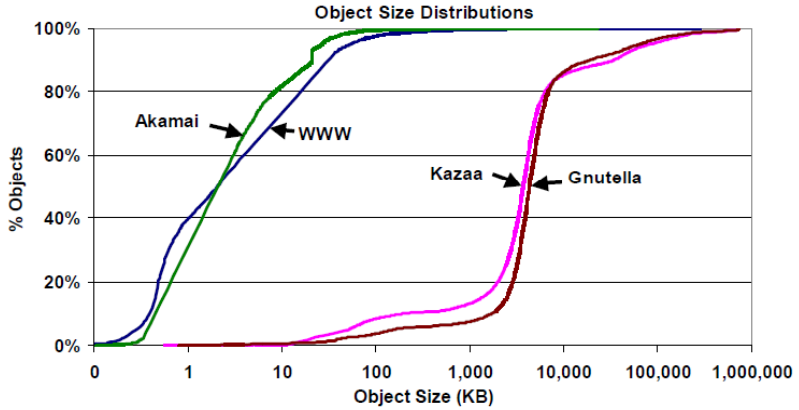
The only constant is (rapid) change …

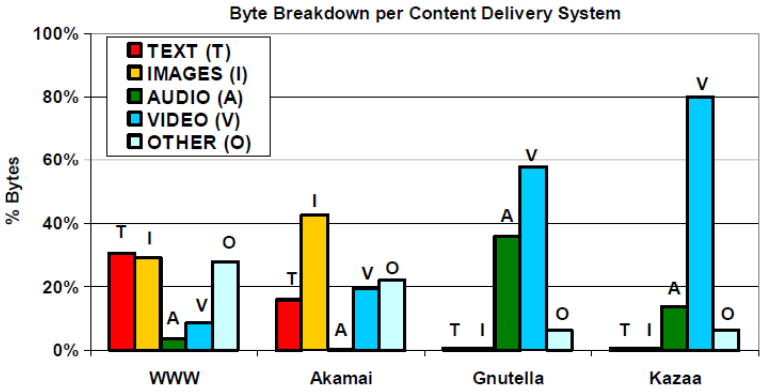
- The rise and fall of email, FTP, Web, P2P, video, …
- Plus trends you can't see, e.g., Skype, Facebook, …



Change shifts traffic features too

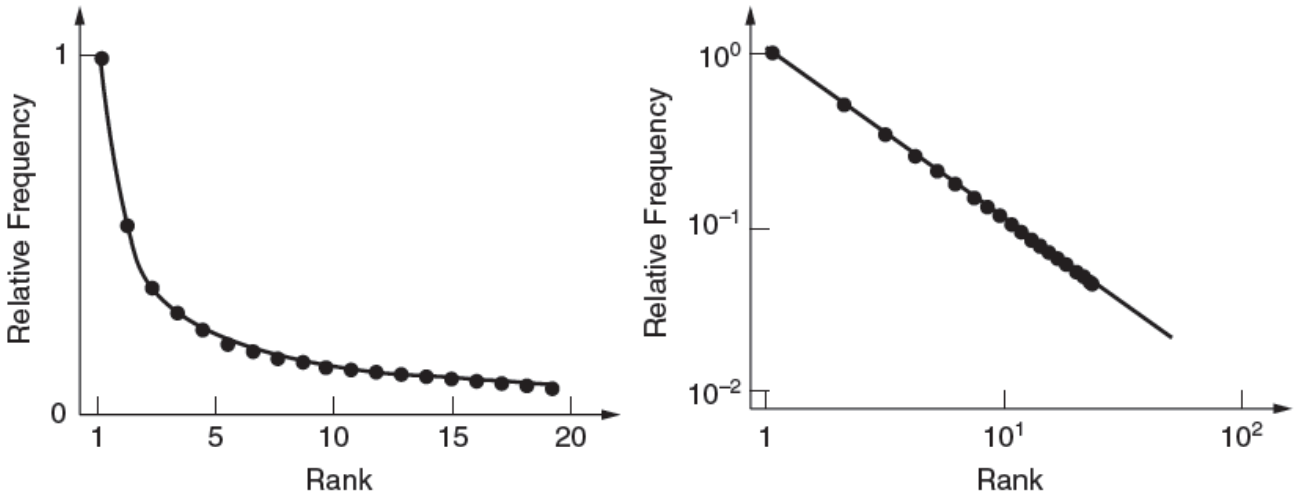
- With the rise of video, shift from many short connections to much longer/larger transfers
- This data from 2002 when P2P exploded





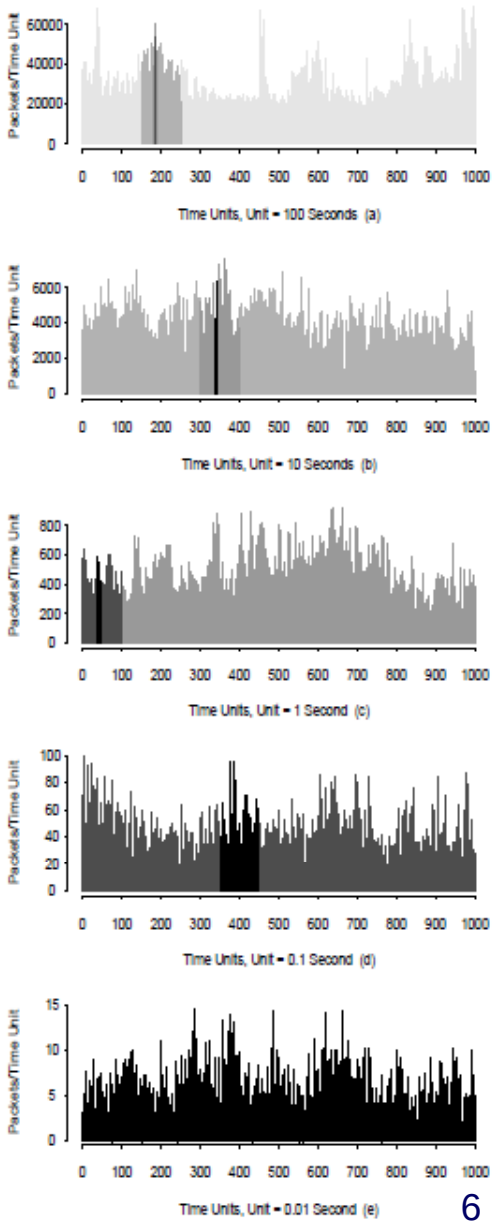
More on macro-level trends of objects

- The size of transfers is heavy-tailed
  - Mostly small connections yet most bytes in a few large ones
- The popularity of objects has a power-law distribution
  - Zipf/Pareto for Web pages



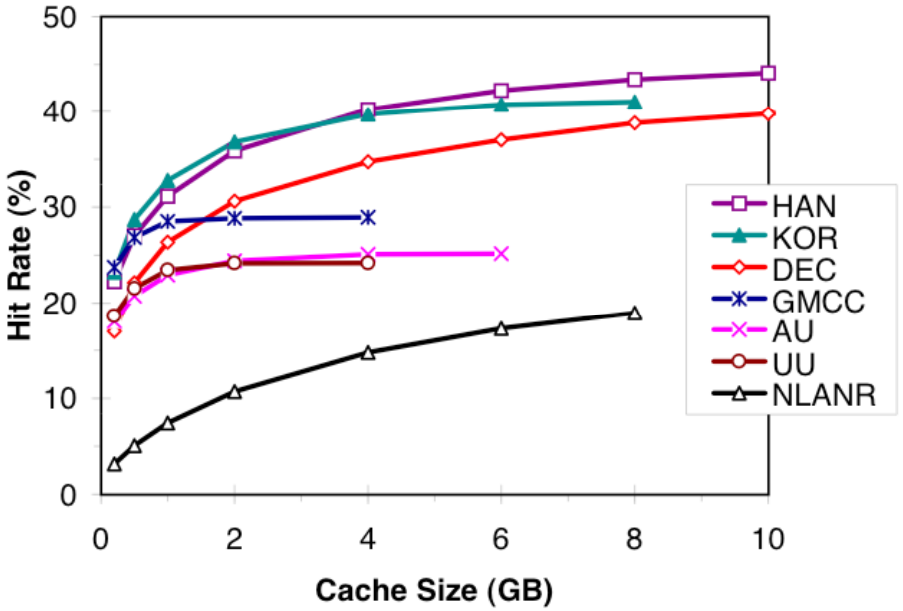
Self-similarity (packet arrivals)

- Network traffic is bursty over all timescales; Poisson is only a good model for human-driven
  - "On the self-similar nature of Ethernet traffic," Leland et al., 1993
- Aggregating Poisson traffic (exponential inter-arrival times) smoothes it
- But aggregating self-similar traffic just makes it burstier



Implications for Caching

- Popular traffic can be cached just fine, but not the tail
  - More like “50%” than “95%”
  - One rule of thumb: hit rate grows as the log of the cache size



(Web proxy cache results)

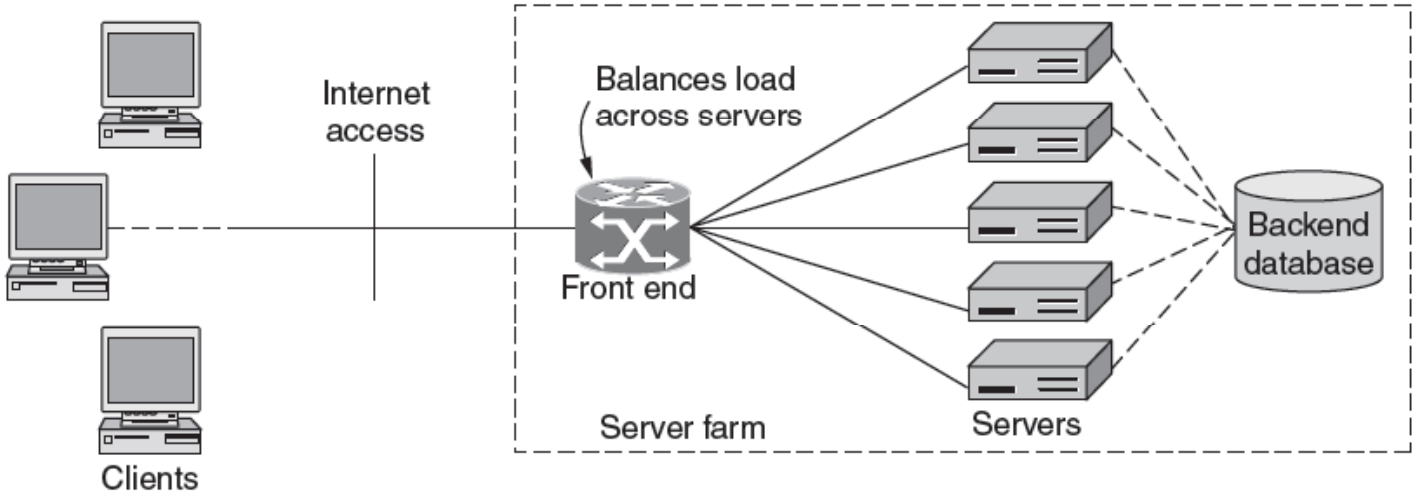
How to speed up content distribution

- Model is that clients request objects from fixed universe, e.g., Web pages, movies

1. Cache client requests
   - Done with browser and proxy caches
2. Remove server bottleneck
   - Replicate it
3. Place content close to clients
   - Reduces network load, speeds transfers (TCP effects etc.)

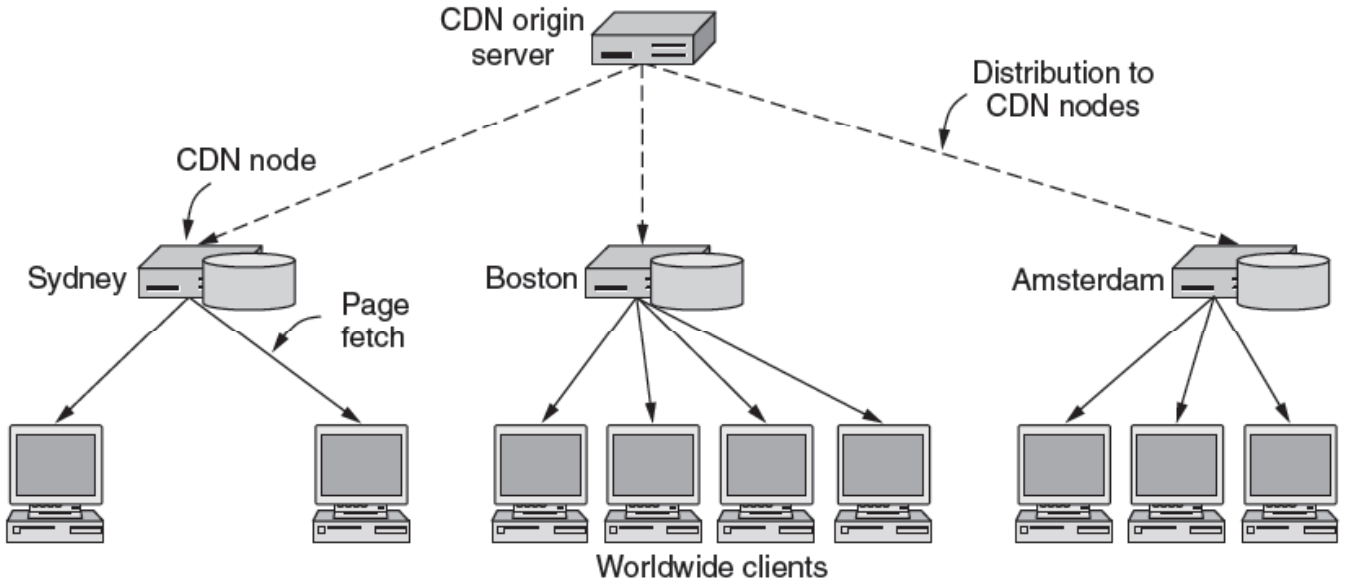
Server farms

- One logical server is really a cluster of machines
  - But there are bandwidth limits as well



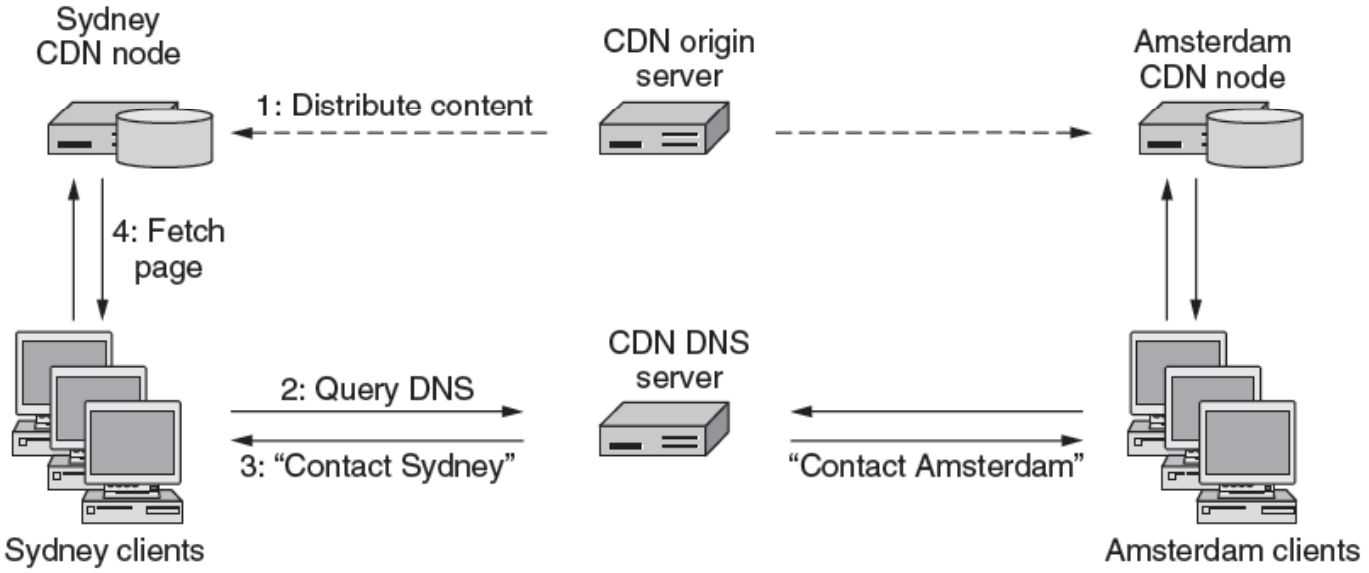
Content Distribution Networks (CDNs)

- Akamai as example. Replicate content at locations near clients; replicas are caches.
- Q: How do clients find them?



CDN operation

- Magic is to override DNS resolution for deployment
  - Client unchanged, uses URL to get page
  - DNS maps name in URL to IP of nearby replica (different answers!)
  - Nearby might be RTT to client nameserver, or better

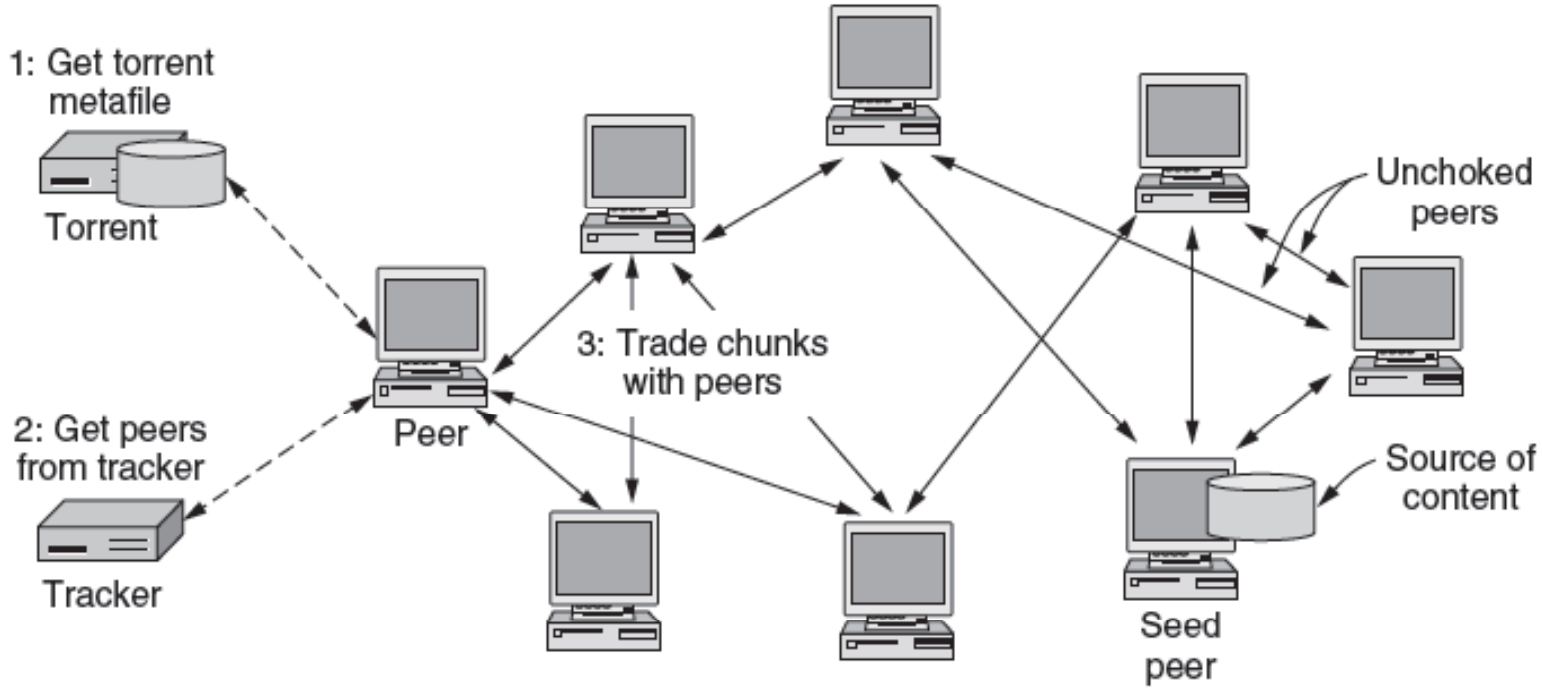


Peer-to-Peer as alternative architecture

- CDNs rely on central administration. P2P is fully distributed ("a group of users helping themselves")

- Users serve dual role as replicas for each other
  - Issues of participation incentives
- Magic is to connect client with a set of nearby replicas
  - Application search process that favors better/faster partners
  - Emphasis on decentralization; no single authority or contact

BitTorrent



- Hang on – isn't the metafile a centralized step? What can we do about that … (Vuze)

Chord – question on DNS

- You could literally do it, but unlikely to be a win
- Advantages:
  - No special root servers
- Disadvantages:
  - No geographic query locality yet (e.g., may need to go to Australia to resolve UW query)
  - Still need a way to divide the namespace for different organizations to use
  - Participation incentives are unclear (if you're a company you don't necessarily need to put in a server node)
  - Reliability and security are unclear (failure or compromise of a node in one company may effect another, unrelated company)

Chord – question on contribution

- Not only O(log N), but fully decentralized.

- This means:
  - No small number of nodes can be shot to make the system fail
  - No small number of nodes carry a disproportionately large load
  - All nodes operate concurrently, without complex locking

Chord – question on content distribution

- Many DHT/P2P scenarios use “user-contributed” nodes. But who provides the infrastructure is somewhat orthogonal to the organization of the nodes.

- This means:
  - Can use DHTs inside large data centers, e.g. Amazon Dynamo, and in fact this simplifies many issues (incentives, security)
  - More large systems are likely to go this way (my guess)

- But there is much more to content distribution, e.g., caching, serving large objects, tracking, access rights, …