6.1800 Spring 2025

Lecture #13: The application layer

serving content as content evolves

Katrina LaCurts | lacurts@mit.edu | 6.1800 2025

6.1800 in the news

standardizations let us communicate across machines

TCP, DNS, OSPF (link-state routing), etc., all have standards that describe the protocols in detail including packet formats

First Look: New Apple Emojis in iOS 18.4 Beta 2

Today Apple has previewed its latest batch of Unicode emojis as part of the second iOS 18.4 beta.

Keith Broni
Mar 3, 2025 · 3 min read

https://blog.emojipedia.org/first-look-new-apple-emojis-in-ios-18-4-beta-2/

6.1800 in the news

who sets the standards?

read this comic and let my friend keith explain it to you!

https://thenib.com/who-makes-emoji/

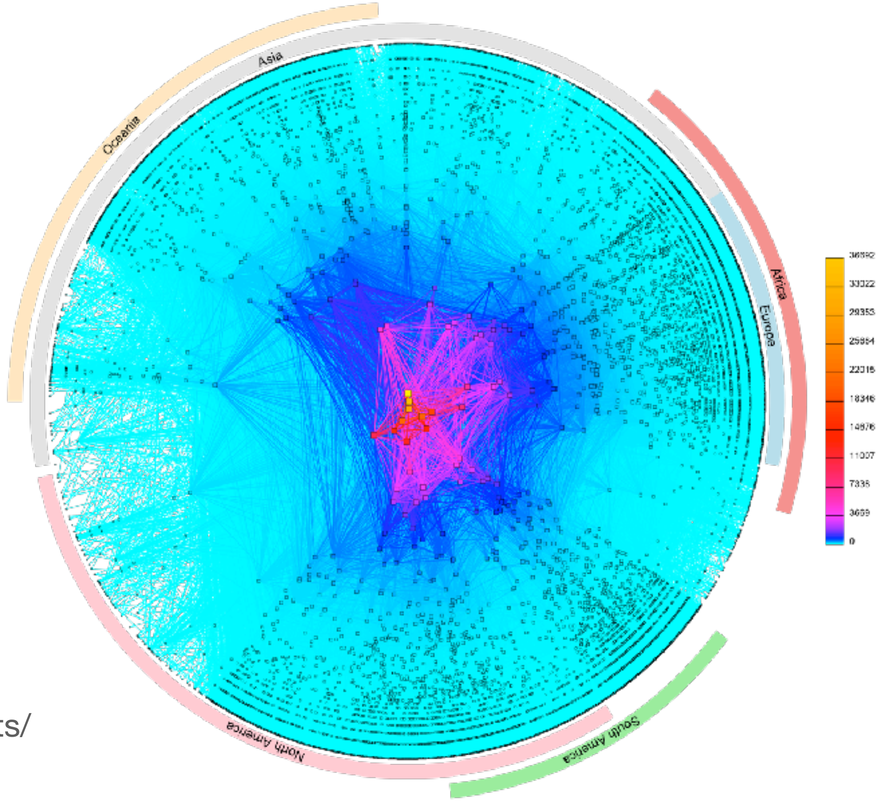
First Look: New Apple Emojis in iOS 18.4 Beta 2

Today Apple has previewed its latest batch of Unicode emojis as part of the second iOS 18.4 beta.

Keith Broni
Mar 3, 2025 · 3 min read

1970s: ARPAnet | 1978: flexibility and layering | early 80s: growth → change | late 80s: growth → problems | 1993: commercialization

hosts.txt | distance-vector routing | TCP, UDP | OSPF, EGP, DNS | congestion collapse (which led to congestion control) | policy routing | CIDR



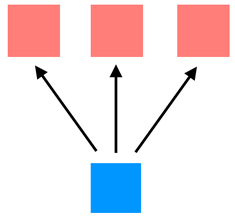
CAIDA's IPv4 AS Core, January 2020 (https://www.caida.org/projects/cartography/as-core/2020/)

today: how do all of the lower layers affect application-layer protocols? specifically, how do we **deliver content** on the Internet?

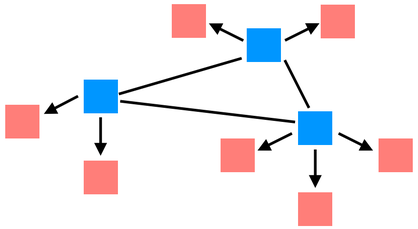
application	the things that actually generate traffic
transport	sharing the network, reliability (or not) *examples: TCP, UDP*
network	naming, addressing, routing *examples: IP*
link	communication between two directly-connected nodes *examples: ethernet, bluetooth, 802.11 (wifi)*

how do we share a file — or **deliver content** — on the Internet?

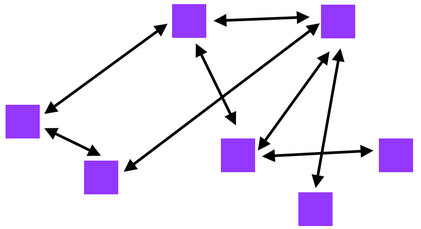
client-server



CDNs



P2P

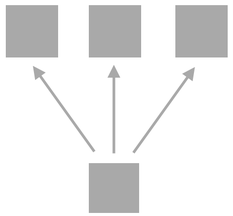


more distributed
more scalable?

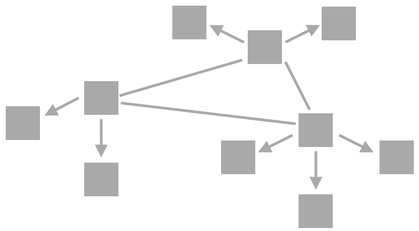
we know that **a client-server model is (relatively) simple, but doesn't scale well**; let's understand more about the other two technologies, to see where they end up in terms of complexity, scalability, etc.

as part of this endeavor, we'll also see why the underlying network matters in these designs

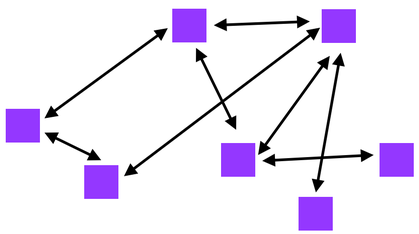
client-server



CDNs



P2P



.torrent file

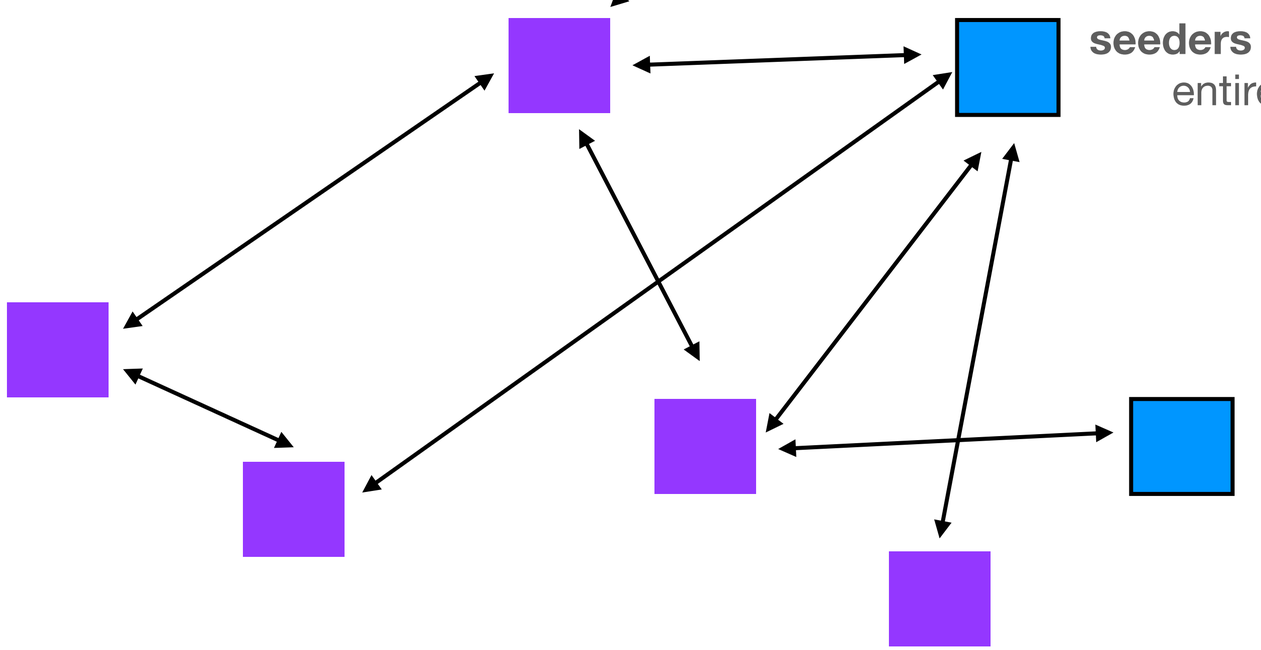
```
file name
file size
information about the
“blocks” of the file
tracker URL
```

tracker

```
list of
peers
```

2. contact tracker for list of peers

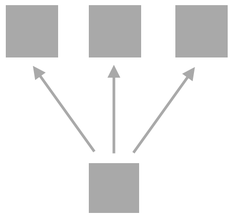
1. download .torrent file from known website



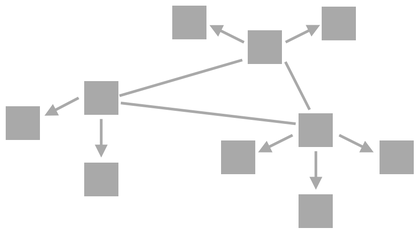
seeders have the entire file

3. communicate with (some) peers to download and upload blocks

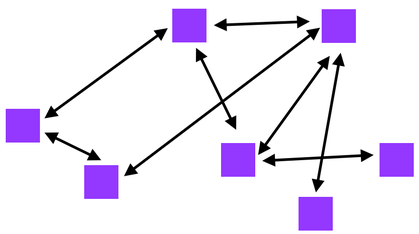
client-server



CDNs



P2P



.torrent file

```
file name
file size
information about the
“blocks” of the file
tracker URL
```

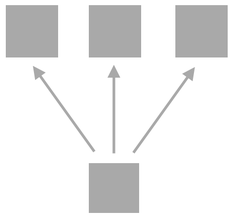
tracker

```
list of
peers
```

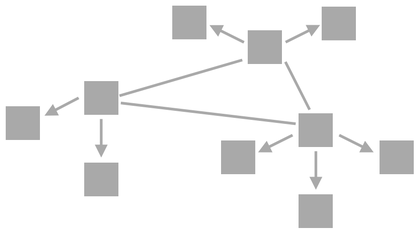
question: are there any **incentives** for peers to upload data to another peer? are there any **drawbacks**?

seeders have the entire file

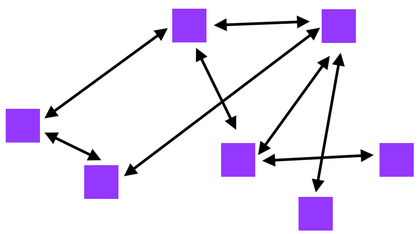
client-server



CDNs

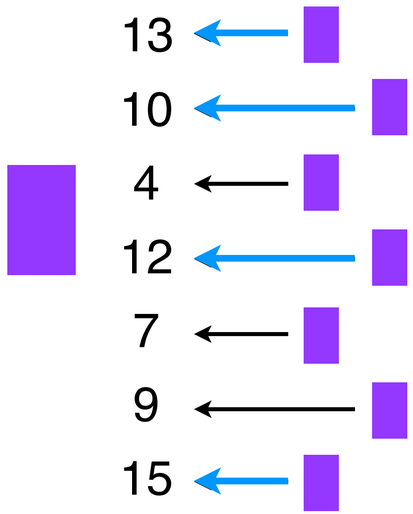


P2P

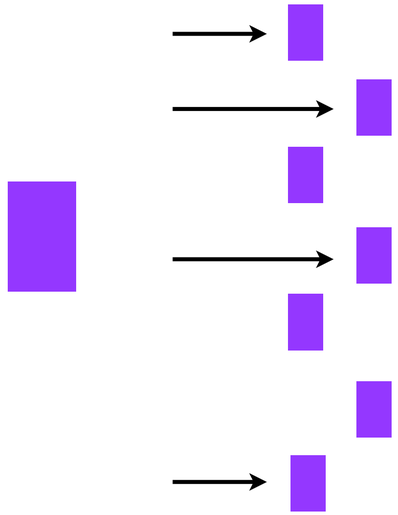


how do we incentivize users to upload?

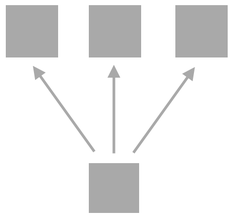
round *t*



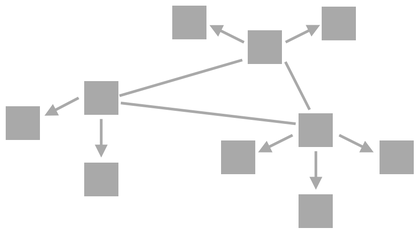
round *t+1*



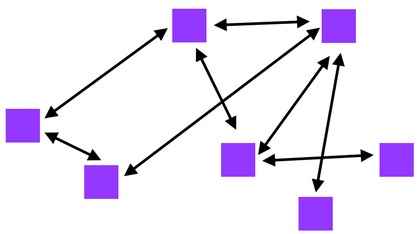
client-server



CDNs



P2P

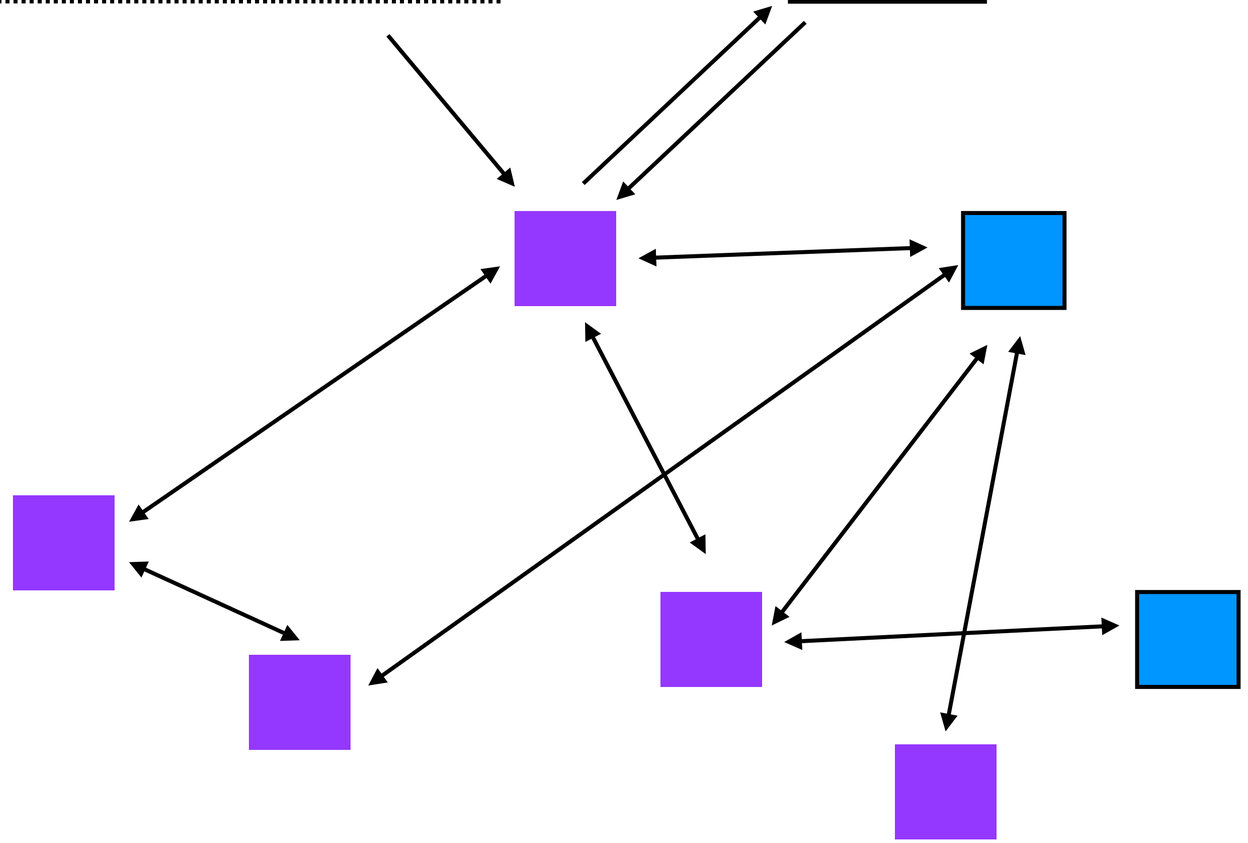


.torrent file

```
file name
file size
information about the
“blocks” of the file
tracker URL
```

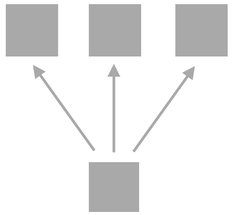
tracker

```
list of
peers
```

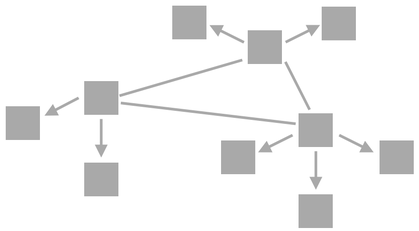


question: are there any **central points of failure** in this network?

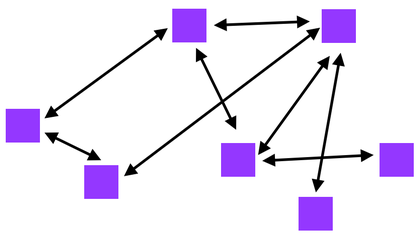
client-server



CDNs

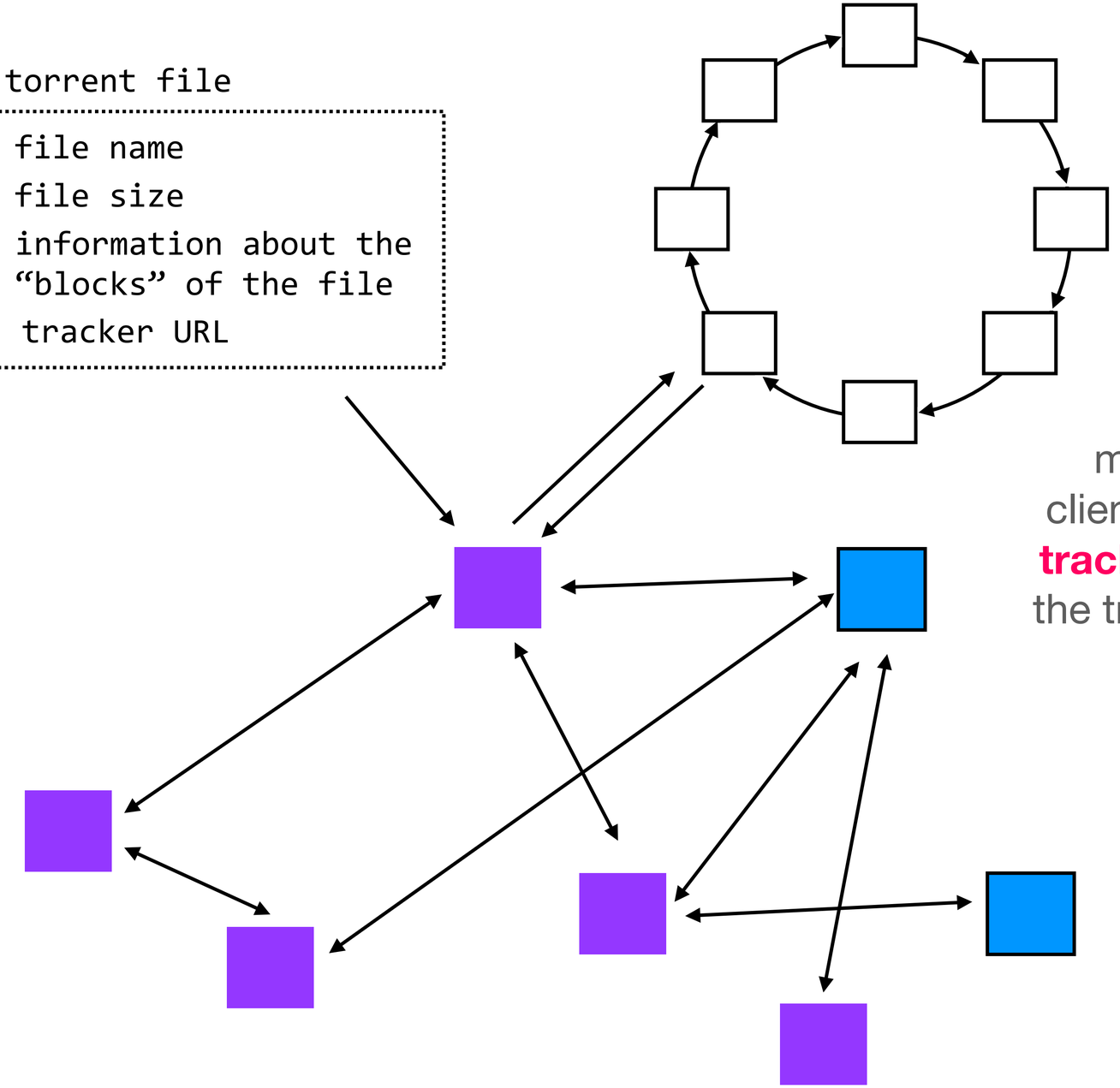


P2P



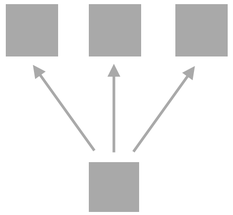
.torrent file

```
file name
file size
information about the
“blocks” of the file
tracker URL
```

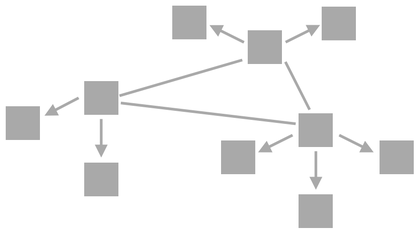


most modern BitTorrent clients used a **decentralized tracker**, where no machine in the tracker network knows the full set of peers

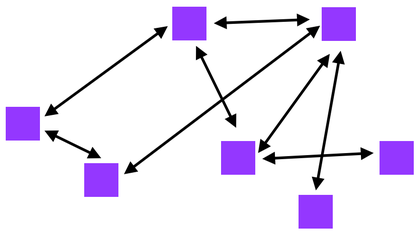
client-server



CDNs

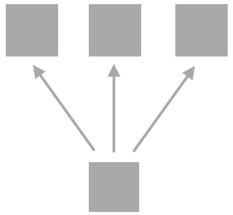


P2P

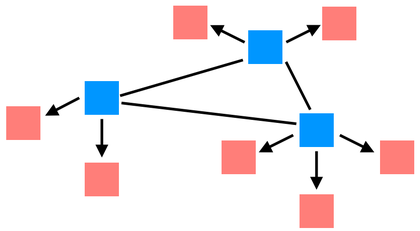


requires some specific organization of the content (e.g., well-defined “blocks”), the ability to discover other peers, and some incentives to get users to upload. in practice, scalability is limited by end-users’ upload constraints

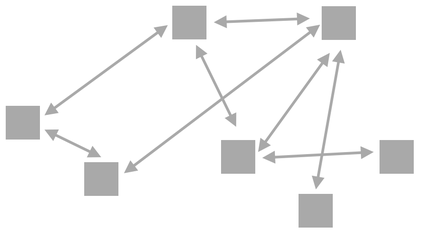
client-server



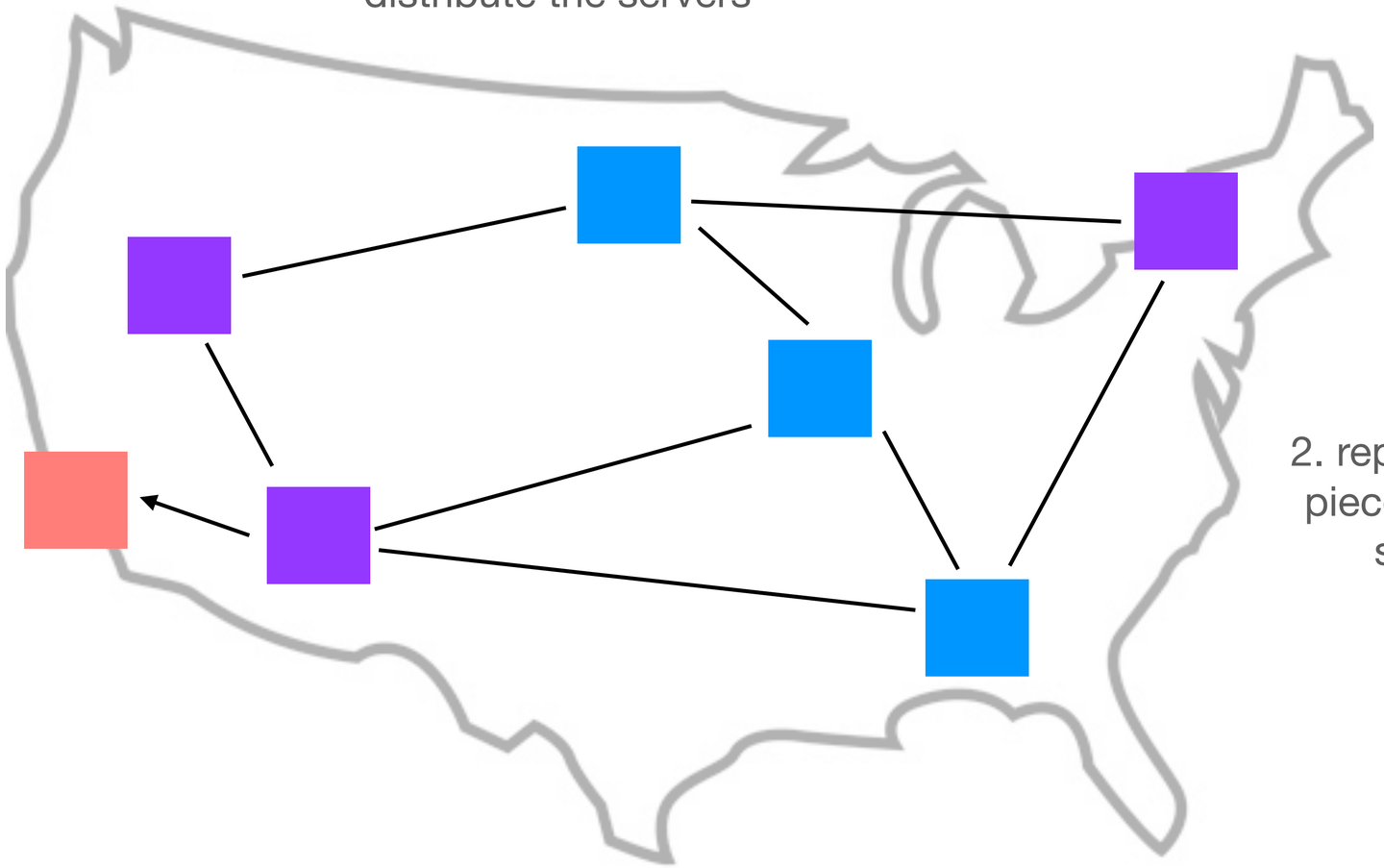
CDNs



P2P



1. geographically distribute the servers

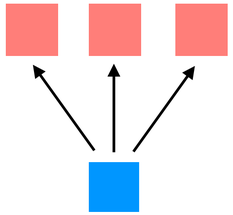


2. replicate a particular piece of content ***p*** on some of them

3. when a client requests ***p***, direct them to the “best” server that has a copy of ***p***

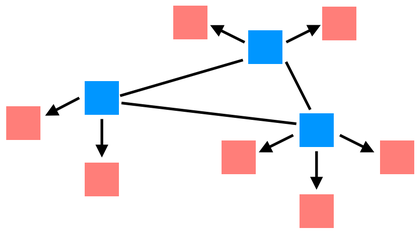
question: what do you think makes a server the “**best**”?

client-server



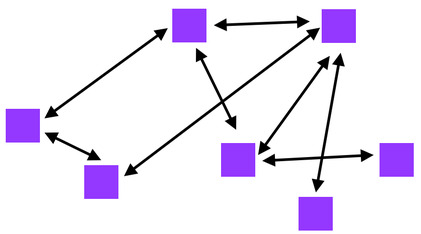
doesn't scale well, but a lot (*a lot*) less complicated than CDNs!

CDNs



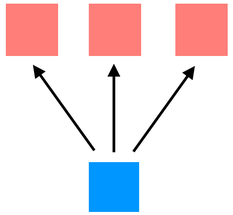
requires a great deal of coordination and organization among the edge servers (all of which are owned by a single company). not as "organic" as P2P networks, but can provide better performance guarantees, in part by finding alternate routes and improving transport-layer performance

P2P

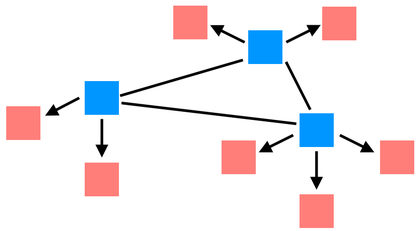


requires some specific organization of the content (e.g., well-defined "blocks"), the ability to discover other peers, and some incentives to get users to upload. in practice, scalability is limited by end-users' upload constraints

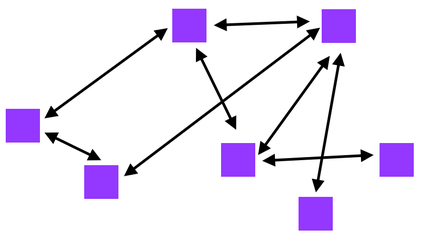
client-server



CDNs



P2P



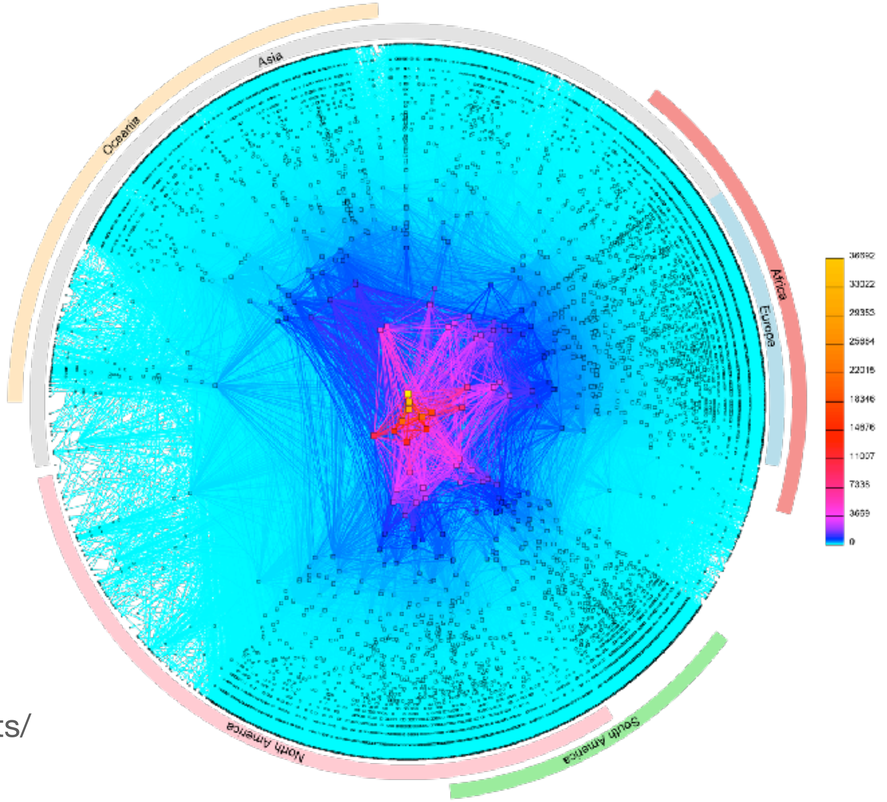
the technologies for sharing content on the Internet have changed as the way we use the Internet has changed

the underlying network affects how well these technologies work, and there are also interesting challenges in terms of how to keep data up-to-date and **consistent** across multiple machines, and how to deal with **failures**

these are challenges we'll address starting after spring break

1970s: ARPAnet | 1978: flexibility and layering | early 80s: growth → change | late 80s: growth → problems | 1993: commercialization

hosts.txt | distance-vector routing | TCP, UDP | OSPF, EGP, DNS | congestion collapse | policy routing | CIDR



CAIDA's IPv4 AS Core, January 2020 (https://www.caida.org/projects/cartography/as-core/2020/)

on the Internet, we have to solve all of the "normal" networking problems (addressing, routing, transport) **at massive scale, while supporting a diverse group of applications and competing economic interests**

application	the things that actually generate traffic
transport	sharing the network, reliability (or not) *examples: TCP, UDP*
network	naming, addressing, routing *examples: IP*
link	communication between two directly-connected nodes *examples: ethernet, bluetooth, 802.11 (wifi)*