

Asymptotics in the Internet:

Three Random Results

Scott Shenker

ACIRI/ICSI

joint work with Lee Breslau, Graham Phillips, Dick Karp, Hongsuda Tangamunrunkit and many others. . .

Scaling: Two Interpretations

Scaling as an interesting phenomenon:

- Structure at all length scales
- Results in self-similarity, long-range dependence, power laws, etc.

Scaling as something to cope with:

- Internet is getting bigger, faster, more diverse
- What happens as it grows?
 - *Descriptive* approach
- How do you design the architecture to ensure that the network continues to function as it grows?
 - *Prescriptive* approach
 - An attempt to *avoid* interesting behavior

Outline

1. Increasing size

- How do multicast trees scale?
- *Descriptive*

2. Increasing size and diversity

- How do we keep the network stable?
- *Prescriptive*

3. Increasing bandwidth

- A design decision: Best-Effort vs Reservations
- *Descriptive and Prescriptive*

Unifying Theme

?

Topic #1

As network grows in size

- how do multicast trees scale?

What is Multicast?

Unicast:

- Packets go from source to single destination
- If you want the same data to reach multiple recipients, you have to send a separate packet to each receiver
- Massive waste of bandwidth when receiver set is large

Multicast:

- Packets sent to multiple destinations at once
 - set of receivers is called the multicast group
- Packets travel down delivery tree
 - e.g., union of unicast paths
- Packets get duplicated at branch points
 - No wasted bandwidth!

Question

Question:

- How much bandwidth does multicast save in the limit of large groups?

Caveat:

- Multicast has many other advantages besides this bandwidth savings

Notation

Tree Size: $L(n)$

- Given network and source
- Place n receivers at random locations uniformly distributed in network
- $L(n)$ is the number of links in the delivery tree reaching these n locations
- \hat{u} is average unicast path length (normalization)

Unicast:

- $L(n) \approx \hat{u}n$

Question

How does $L(n)$ behave for large n ?

- $M \gg n \gg 1$ with M being the size of the network
- The deviations from linearity quantify the bandwidth savings of multicast
- This work was inspired by the (surprising) results of Chaung and Sirbu

k-ary Trees with Receivers at Leaves

Notation:

- D is depth of tree ($\hat{u} = D$)
- $M = k^D$ is total number of leaves

Formula:

$$L(n) = \sum_{l=1}^D k^l \left(1 - (1 - k^{-l})^n\right)$$

Approximation: (when $n < M$)

$$L(n) \approx n \left(c - \frac{\ln \frac{n}{M}}{\ln k} \right)$$

Extension:

- Adding receivers throughout merely changes the constant c

Clustering

In reality, receivers are not uniformly distributed:

- Does clustering destroy our results?

Model:

- Place receivers uniformly
- Weigh the resulting configurations based on the average inter-receiver distance \hat{d}
- $W = e^{-\beta\hat{d}}$

Parameter β :

- $\beta > 0$ emphasizes clustered configurations
- $\beta < 0$ emphasizes spread out configurations

Summary of Clustering Results

- Clustering certainly affects the value of $L(n)$
- In the limit of large M with fixed $\frac{n}{M}$ the asymptotic nature of $\frac{L(n)}{n}$ is unchanged by clustering.

Realistic Networks

What happens in more realistic networks?

We measured $L(n)$ for several realistic topologies:

Real

name	date	nodes	links	avg. degree
ARPA	Early	47	68	2.9
MBONE	Feb 1999	4179	8529	4.1
Internet	March 1999	56317	77154	2.7
AS	March 1999	4830	9077	3.8

Generated

name	source	nodes	links	avg. degree
r100	GT-ITM (random)	100	177	3.5
ts1000	GT-ITM (transit-stub)	1000	1819	3.6
ts1008	GT-ITM (transit-stub)	1008	3787	7.5
ti5000	TIERS	5000	7084	2.8

Oversimplified Results for Realistic Networks

Three Points:

- Depends on the behavior of $T(r)$
 - The number of nodes within r hops of the source
 - $T(r) \approx e^{\lambda r}$ for a tree or random network
 - $T(r) \approx r^\alpha$ for a mesh
- Asymptotic form holds when $T(r)$ is exponential

$$L(n) \approx n \left(c - \frac{\ln \frac{n}{M}}{\lambda} \right) \text{ for some constants } c, \lambda$$

- $T(r)$ is exponential for all networks except for Tiers and MBone
 - Tiers and MBone have slower growth, but faster than a mesh-like power law

Open Questions

- Do “real” networks have exponential $T(r)$?
 - why or why not?
 - disagreement with Faloutsos³ conclusion
- Why are the Tiers and MBone networks different from the others?
 - MBone is explicitly configured
 - The top level of the hierarchy in Tiers is optimized
- Which dominates in network design, and why?
 - randomness, hierarchy, or geometry

Metaquestion

Does the asymptotic limit mean anything in human-engineered systems?

- Network growth does not follow natural laws, but instead results from human decisions at various scales
- These decisions depend on factors that can change radically
 - economic, legal, social, technical, . . .

Topic #2

As networks becomes bigger, faster, and more diverse

- How do we ensure that the network remains stable?

Congestion Control: the Reader's Digest Version

Sally Floyd's talk will be the unabridged version

Congestion:

- If flows overload a link, delays increase and packets are dropped
 - we call this *congestion*
 - all flows suffer performance degradation

Controlling congestion:

- Flows detect the congestion through packet drops or (soon) ECN bits
- Flows respond to congestion by reducing their sending rate
- Flows respond to the absence of congestion by increasing their sending rate

Congestion Control as a Dynamical System

Dynamical system:

- Flows individually vary their rates in response to collectively induced congestion

One (of many) design goals:

- The steady states of the system should be (linearly) stable
 - This is stability of the system as a whole, not just for an individual flow
- This should be true for *all* network configurations

Simple Model

Routers:

- Router α produces congestion signal b_α
- Function of average queue size at α

Flows:

- Send at rate r_i
- Experience maximal congestion along path:

$$b_i = \max_{path} b_\alpha$$

- Adjust rate according to:

$$r_i \leftarrow r_i + f(r_i, b_i, d_i)$$

Stability Goal

System-Wide Stability (SWS):

- All fixed points are linearly stable, for all networks
- Eigenvalues of $I + \frac{df_i}{dr_j}$ have magnitude ≤ 1

Two Achievable Subgoals

Time-scale-invariance (TSI):

- If all links speeds double, the steady state rates r_i should double

Unilateral stability (US):

- At fixed point, want $-1 \leq 1 + \frac{df}{dr_i} \leq 1$
- Want this to be true under *all* conditions

Serendipity

Theorem:

- TSI and US \Rightarrow SWS

Comment:

- An example of how making the architecture scalable does not involve *scaling* or other interesting behaviors
- Boring is better!

Open Question:

- Does this result apply to more realistic models?
- Model is currently a *fairy tale*
 - nice moral, little reality...

Topic #3

As bandwidth grows, which architecture is best?

- Best Effort or Reservations

Best Effort

Offered by current Internet:

- Network does its best to deliver packets
- No assurances about when, or even whether, packets will be delivered
- No admission control:
 - flows can send when they want
 - they get share of currently available bandwidth

Reservations

Proposed extension to Internet architecture:

- Allows flows to reserve bandwidth (if they want)
- These flows receive their requested bandwidth
- Admission control for flows wanting reservations
 - these flows need to ask for reservation
 - can be turned away if network is too congested

Ongoing Debate: Best Effort versus Reservations

Reservation advocates:

- Best-effort service fine for data applications
- Real-time applications need reservations

Best-Effort-Only advocates:

- Best-Effort is good enough for all applications
- Adaptive real-time applications can cope with any level of service
- Won't need reservations as network speeds increase

My goal:

- Use a simple model to inform the debate
- ... and prove I'm right!

Basic Model

Network:

- A single link
 - Capacity C
 - Shared equally by k flows

Flows:

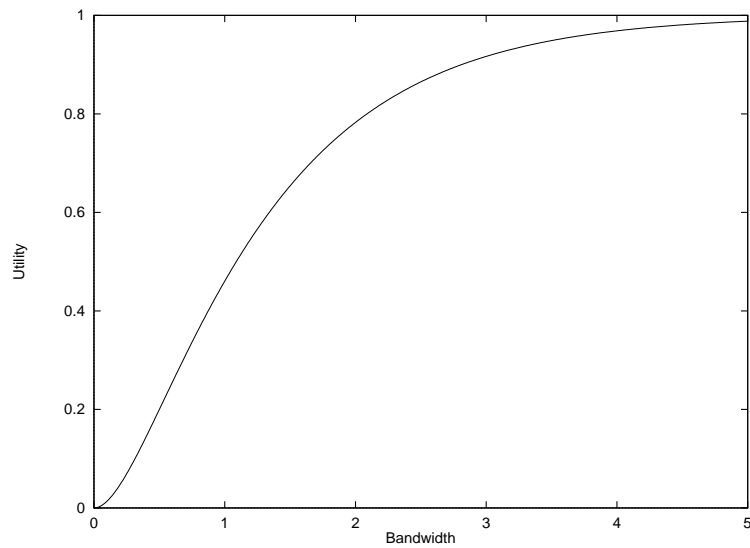
- Utility $\pi(b)$
 - b is bandwidth share
- Total utility: $k\pi(\frac{C}{k})$
 - Let $\tilde{k} = \operatorname{argmax}[k\pi(\frac{C}{k})]$

Utility Functions

Rigid Real-Time:

- $\pi(b) = 1$ for $b \geq 1$
- $\pi(b) = 0$ for $b < 1$

Adaptive Real-Time:



Load Model

Varying load:

- Ignore actual time dependence of load
- Use static probability distribution $P(k)$
- Compute average of total utility

Three cases studied:

- Poisson: $P(k) = \frac{\nu^k e^{-\nu}}{k!}$
- Exponential: $P(k) = (1 - e^{-\beta})e^{-\beta k}$
- Algebraic: $P(k) = \frac{\nu}{\lambda + k^z}$ (for $z > 2$)

Two Architectures

Best-Effort-Only

- Admit all k flows
- Total utility:

$$B(C) = \sum_{k=1}^{\infty} P(k) k \pi\left(\frac{C}{k}\right)$$

Reservations:

- Limit load to $k \leq \tilde{k}$
- Total utility:

$$R(C) = \sum_{k=1}^{\tilde{k}} P(k) k \pi\left(\frac{C}{k}\right) + \sum_{k=\tilde{k}+1}^{\infty} P(k) \tilde{k} \pi\left(\frac{C}{\tilde{k}}\right)$$

Comparison:

- $R(C) \geq B(C)$, but by how much?
- Particularly in the $C \rightarrow \infty$ limit

Quantitative Comparisons

Performance Gap:

- $\delta(C) = R(C) - B(C)$
- Units are somewhat arbitrary

Bandwidth Gap:

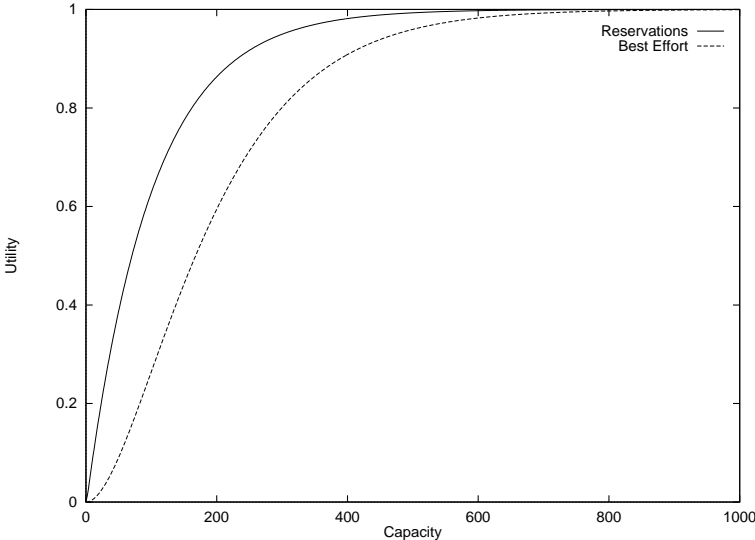
- $\Delta(C)$ where $R(C) = B(C + \Delta(C))$
- How much additional capacity does a best-effort-only network need to achieve the same utility as a reservation-capable network?

Architectural choice is bandwidth versus complexity:

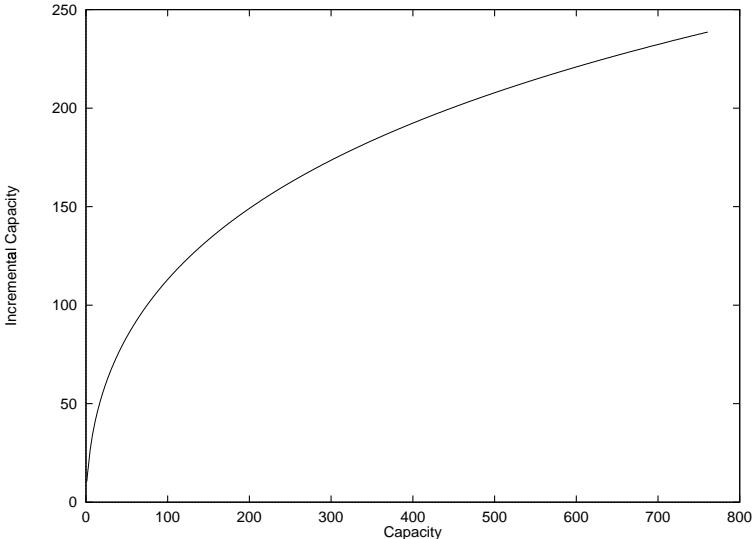
- Can't quantify architectural complexity
- $\Delta(C)$ describes the bandwidth gain

Example: Exponential Load, Rigid Applications

$R(C)$ and $B(C)$



Bandwidth Gap $\Delta(C)$



Summary of Asymptotic Results

Performance Gap: δ

- Significant for $C \approx \langle k \rangle$
- For $C \rightarrow \infty$, quickly converges to zero

Bandwidth Gap: Δ

- Poisson load: $\Delta \rightarrow 0$
- Exponential/adaptive: $\Delta \rightarrow 0$
- Exponential/rigid: $\Delta \approx \ln C$
- Algebraic load: $\Delta \propto C$

Bound:

- $\lim_{C \rightarrow \infty} \frac{\Delta(C)}{C} \leq (e - 1)$
- Realized in $z \rightarrow 2^+$ limit

Oversimplified Lessons from Model

No definitive verdict:

- Best-effort seems sufficient with Poisson and Exponential load models
 - assuming one can overprovision: $C \gg \langle k \rangle$
- Reservations provide some benefit with Algebraic load
 - without extensions to model this benefit is bounded
 - with extensions this benefit is not bounded, but not clear what the typical values are

To advocate reservations, must argue:

- Networks will operate in the regime of $C \approx \langle k \rangle$,
or
- Load is algebraic with $z \approx 2$

Open Question

Will the future Internet load be algebraic?

- If so, where does this power law come from?

Closing Comments

- Is there any unifying theme to Internet asymptotics?
- Is there any unifying theme to making the Internet architecture scalable?
- Does making the Internet architecture scalable have anything to do with power laws, self-similarity, etc.?
 - In the HOT lexicon, are we in the robust or optimized regime?