Congestion Control and Fairness in Named Data Networks

Edmund Yeh

Joint work with Ying Cui, Ran Liu, Tracey Ho

Electrical and Computer Engineering

Northeastern University

NDN Retreat

March 21, 2016

Overview

- NDN enables full utilization of bandwidth and storage.
- Focus on user demand rate for content satisfied by network, rather than session rates.
- General VIP framework for caching, forwarding and congestion control.
- Distributed caching, forwarding, congestion control algorithms which maximize aggregate utility subject to network layer stability.
- VIP congestion control enables fairness among content types.
- Experimental results: superior performance in user delay, rate of cache hits, utility-delay tradeoff.

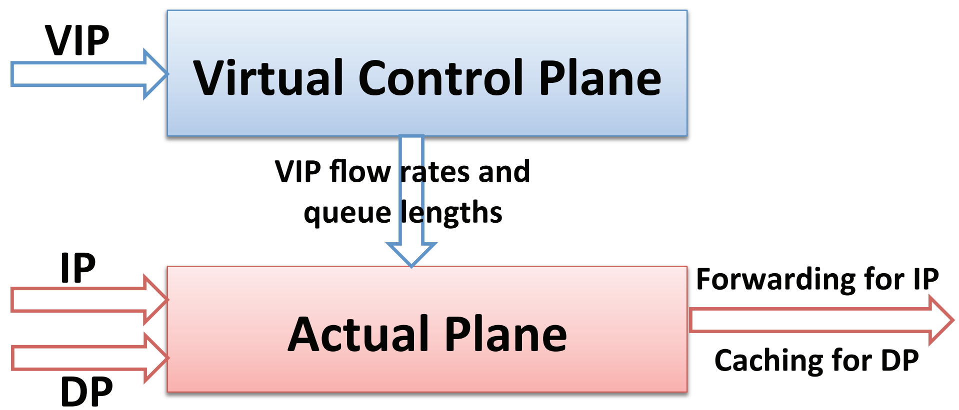
Network Model

- General connected network with bidirectional links and set of caches.
- Each node n aggregates many network users.
- Content in network identified as set $\mathcal{K}$ of data objects.
- For each data object k, there is set of content source nodes.
- IPs for given data object can enter at any node, exit when satisfied by matching DP at content source, or at caching points.
- Content sources fixed, while caching points may vary in time.
- Assume routing (topology discovery and data reachability) already done: FIBs populated for various data objects.

.

Virtual Interest Packets and VIP Framework

- For each interest packet (IP) for data object k entering network, generate 1 (or c) corresponding VIP(s) for object k.
- IPs may be suppressed/collapsed at NDN nodes, VIPs are not suppressed/collapsed.
- VIPs represent locally measured demand/popularity for data objects.



- General VIP framework: control and optimization on VIPs in virtual plane; mapping to actual plane.

VIP Potentials and Gradients

- Each node n maintains a separate VIP queue for each data object k.
- VIP queue size for node n and data object k at beginning of time slot t is counter $V_n^k(t)$.
- Initially, all VIP counters are 0. As VIPs are created along with IP requests, VIP counters incremented at entry nodes.
- VIPs for object k removed at content sources and caching nodes for object k: sinks or attractors.
- Physically, VIP count represent potential. For any data object, there is downward gradient from entry points of IP requests to sinks.

Throughput Optimal Caching and Forwarding

- VIP count used as common metric for determining caching and forwarding in virtual and actual control planes.

- Forwarding strategy in virtual plane uses backpressure algorithm.

- Multipath forwarding algorithm; incorporates link capacities on reverse path taken by DPs.

- Caching strategy given by the solution of max-weight knapsack problem involving VIP counts.

- VIP forwarding and caching algorithm exploits both bandwidth and storage resources to maximally balance out VIP load, preventing congestion buildup.

- Both forwarding and caching algorithms are distributed.

VIP Stability Region and Throughput Optimality

- λ_n^k = long-term exogenous VIP arrival rate at node n for object k:

- VIP network stability region Λ = set of all $\boldsymbol{\lambda} = (\lambda_n^k)_{k\in\mathcal{K},n\in\mathcal{N}}$ for which there exist some feasible joint forwarding/caching policy which can guarantee that all VIP queues are stable.

- VIP Algorithm is throughput optimal in virtual plane: adaptively stabilizes all VIP queues for any $\boldsymbol{\lambda} \in \mathrm{int}(\Lambda)$ without knowing $\boldsymbol{\lambda}$.

- Forwarding of Interest Packets in actual plane: forward each IP on link with maximum average VIP flow over sliding window.

- Caching of Data Packets in actual plane: designed stable caching algorithm based on VIP flow in virtual plane.

VIP Congestion Control

- Even with optimal caching and forwarding, excessively large request rates can overwhelm network.

- No source-destination pairs: traditional congestion control algorithms inappropriate.

- Need content-based congestion control to cut back demand rates fairly.

- VIP framework: can optimally combine congestion control with caching and forwarding.

- Hop-by-hop content-based backpressure approach; no concept of flow.

VIP Congestion Control

- Arriving IPs (VIPs) first enter transport layer queues before being admitted to network layer.

- VIP counts relay congestion signal to IP entry nodes via backpressure effect.

- Congestion control: support a portion of VIPs which maximizes sum of utilities subject to network layer VIP queue stability.

- Choice of utility functions lead to various fairness notions (e.g. max-min, proportional fairness).

Utility Maximization Subject to Network Stability

- $\boldsymbol{\theta}$-optimal admitted VIP rate:

$$\begin{aligned} \bar{\boldsymbol{\alpha}}^*(\boldsymbol{\theta}) = \arg\max_{\bar{\boldsymbol{\alpha}}} \quad & \sum_{n \in \mathcal{N}} \sum_{k \in \mathcal{K}} g_n^k\left(\bar{\alpha}_n^k\right) \\ \text{s.t.} \quad & \bar{\boldsymbol{\alpha}} + \boldsymbol{\theta} \in \Lambda \\ & \mathbf{0} \preceq \bar{\boldsymbol{\alpha}} \preceq \boldsymbol{\lambda} \end{aligned}$$

- $g_n^k(\cdot)$: increasing, concave content-based utility functions.
- $\bar{\boldsymbol{\alpha}}$ = IP (VIP) input rates admitted to network layer.
- $\boldsymbol{\theta}$ = margin to boundary of VIP stability region Λ.
- Maximum sum utility achieved at $\overline{\boldsymbol{\alpha}}^*(\mathbf{0})$ when $\boldsymbol{\theta} = \mathbf{0}$.
- Tradeoff between sum utility attained and user delay.

Transport and Network Layer VIP Dynamics

- Transport-layer queue evolution:

$$Q_n^k(t+1) = \min\left\{\left(Q_n^k(t) - \alpha_n^k(t)\right)^+ + A_n^k(t), Q_{n,\max}^k\right\} \tag{1}$$

- Network-layer VIP count evolution:

$$V_n^k(t+1) \leq \left(\left(V_n^k(t) - \sum_{b\in\mathcal{N}} \mu_{nb}^k(t)\right)^+ + \alpha_n^k(t) + \sum_{a\in\mathcal{N}} \mu_{an}^k(t) - r_n s_n^k(t)\right)^+ \tag{2}$$

Joint Congestion Control, Caching and Forwarding

- Virtual queues $Y_n^k(t)$ and auxiliary variables $\gamma_n^k(t)$.
- Initialize: $Y_n^k(0) = 0$ for all k, n.
- Congestion Control: for each k and n, choose:

$$\alpha_n^k(t) = \begin{cases} \min\left\{Q_n^k(t), \alpha_{n,\max}^k\right\}, & Y_n^k(t) > V_n^k(t) \\ 0, & \text{otherwise} \end{cases}$$

$$\begin{aligned} \gamma_n^k(t) = \arg\max_{\gamma} \quad & W g_n^k(\gamma) - Y_n^k(t) \\ s.t. \quad & 0 \leq \gamma \leq \alpha_{n,\max}^k \end{aligned}$$

  where $W > 0$ is control parameter affecting utility-delay tradeoff.

  Based on chosen $\alpha_n^k(t)$ and $\gamma_n^k(t)$, transport layer queue updated as in (1) and virtual queue updated as:

$$Y_n^k(t+1) = \left(Y_n^k(t) - \alpha_n^k(t)\right)^+ + \gamma_n^k(t)$$

- Caching and Forwarding: Same as VIP Algorithm above. Network layer VIP count updated as in (2).

Joint Congestion Control, Caching and Forwarding

- Joint algorithm adaptively stabilizes all VIP queues for any $\boldsymbol{\lambda}$ inside or outside Λ, without knowing $\boldsymbol{\lambda}$.
- Users need not know utility functions and demand rates of other users.

Theorem 3 *For an arbitrary IP arrival rate* $\boldsymbol{\lambda}$ *and for any* $W > 0$,

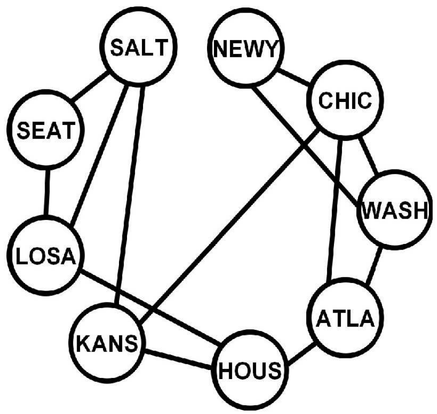
$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=1}^{t} \sum_{n\in\mathcal{N},k\in\mathcal{K}} \mathbb{E}[V_n^k(\tau)] \leq \frac{2N\hat{B} + WG_{\max}}{2\hat{\epsilon}}$$

$$\liminf_{t\to\infty} \sum_{n\in\mathcal{N},k\in\mathcal{K}} g_n^k\left(\overline{\alpha}_n^k(t)\right) \geq \sum_{n\in\mathcal{N},k\in\mathcal{K}} g_n^{(c)}\left(\overline{\alpha}_n^{k*}(\mathbf{0})\right) - \frac{2N\hat{B}}{W}$$

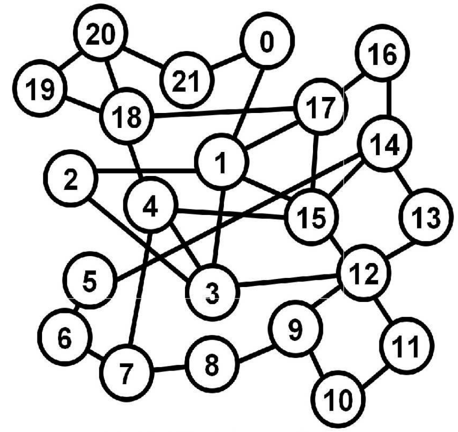
where $\hat{B} \triangleq \frac{1}{2N}\sum_{n\in\mathcal{N}}\Big((\mu_{n,\max}^{out})^2+(\alpha_{n,\max}+\mu_{n,\max}^{in}+r_{n,\max})^2+2\mu_{n,\max}^{out}r_{n,\max}\Big)$,
$\hat{\epsilon} \triangleq \sup_{\{\boldsymbol{\epsilon}:\boldsymbol{\epsilon}\in\Lambda\}} \min_{n\in\mathcal{N},k\in\mathcal{K}}\left\{\epsilon_n^k\right\}$, $\alpha_{n,\max} \triangleq \sum_{k\in\mathcal{K}} \alpha_{n,\max}^k$,
$G_{\max} \triangleq \sum_{n\in\mathcal{N},k\in\mathcal{K}} g_n^k\left(\alpha_{n,\max}^k\right)$, $\overline{\alpha}_n^k(t) \triangleq \frac{1}{t}\sum_{\tau=1}^{t}\mathbb{E}[\alpha_n^k(\tau)]$.

Numerical Experiments

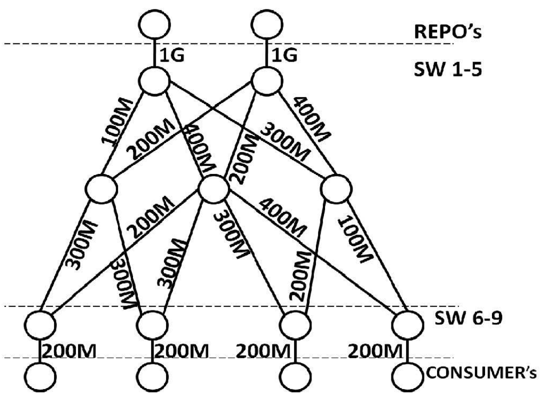
Abilene Topology



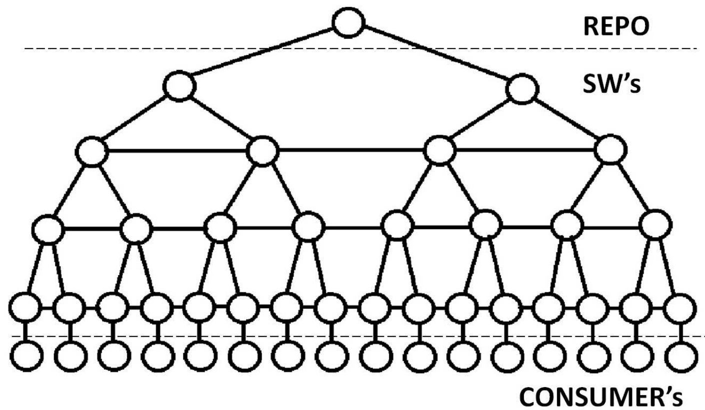
GEANT Topology



Fat Tree Topology



Wireless Backhaul Topology



Network Parameters

- Abilene: 5000 objects, cache size $5GB$ (1000 objects), link capacity 500 Mb/s; all nodes generate requests and can be data sources.

- GEANT: 2000 objects, cache size $2GB$ (400 objects), link capacity 200 Mb/s; all nodes generate requests and can be sources.

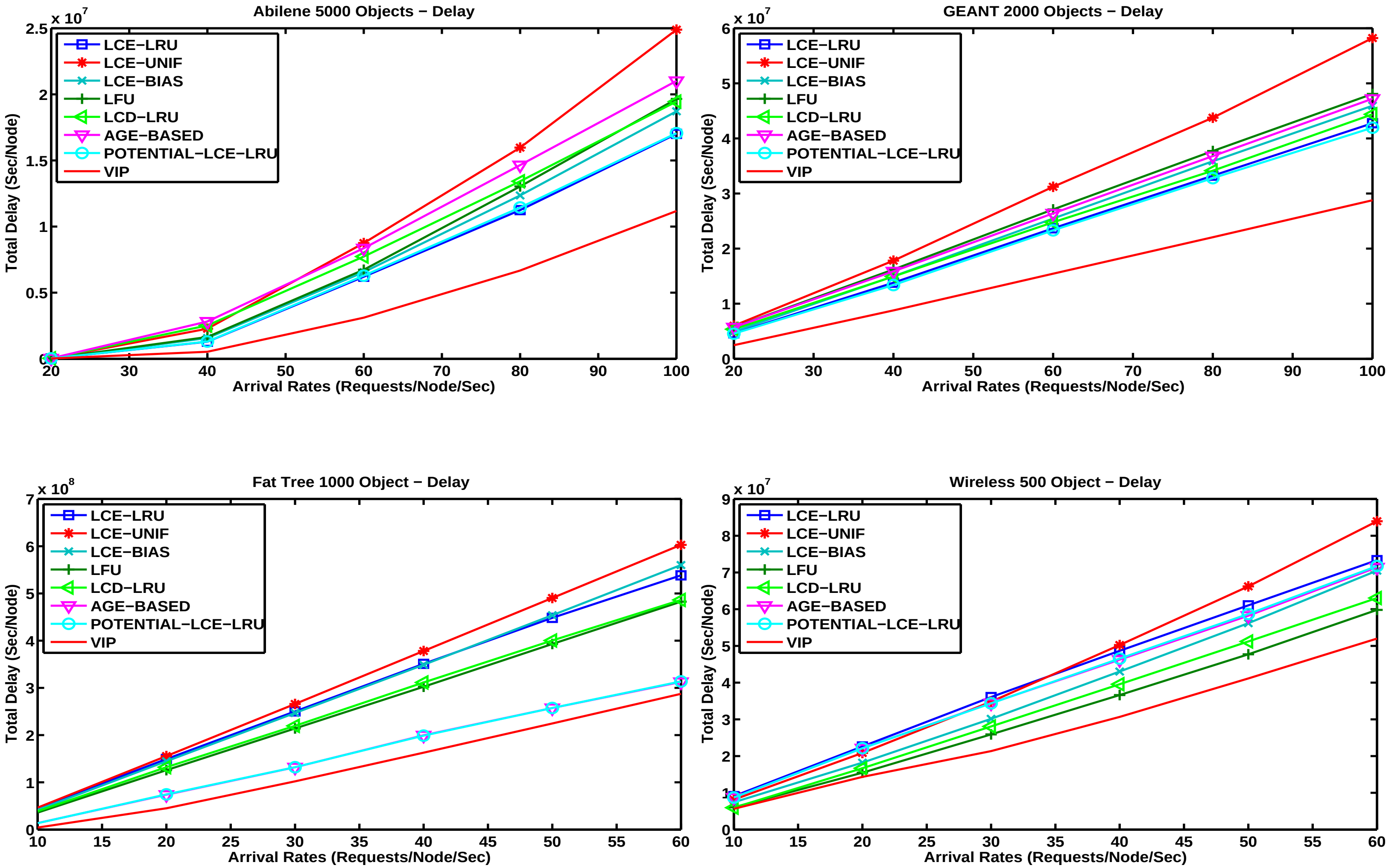
- Fat Tree: 1000 objects, cache size $1GB$ (200 objects); CONSUMER nodes generate requests; REPOs are source nodes.

- Wireless Backhaul: 500 objects, cache size $100MB$ (20 objects), link capacity $500Mb/s$; CONSUMER nodes generate requests; REPO is source node.

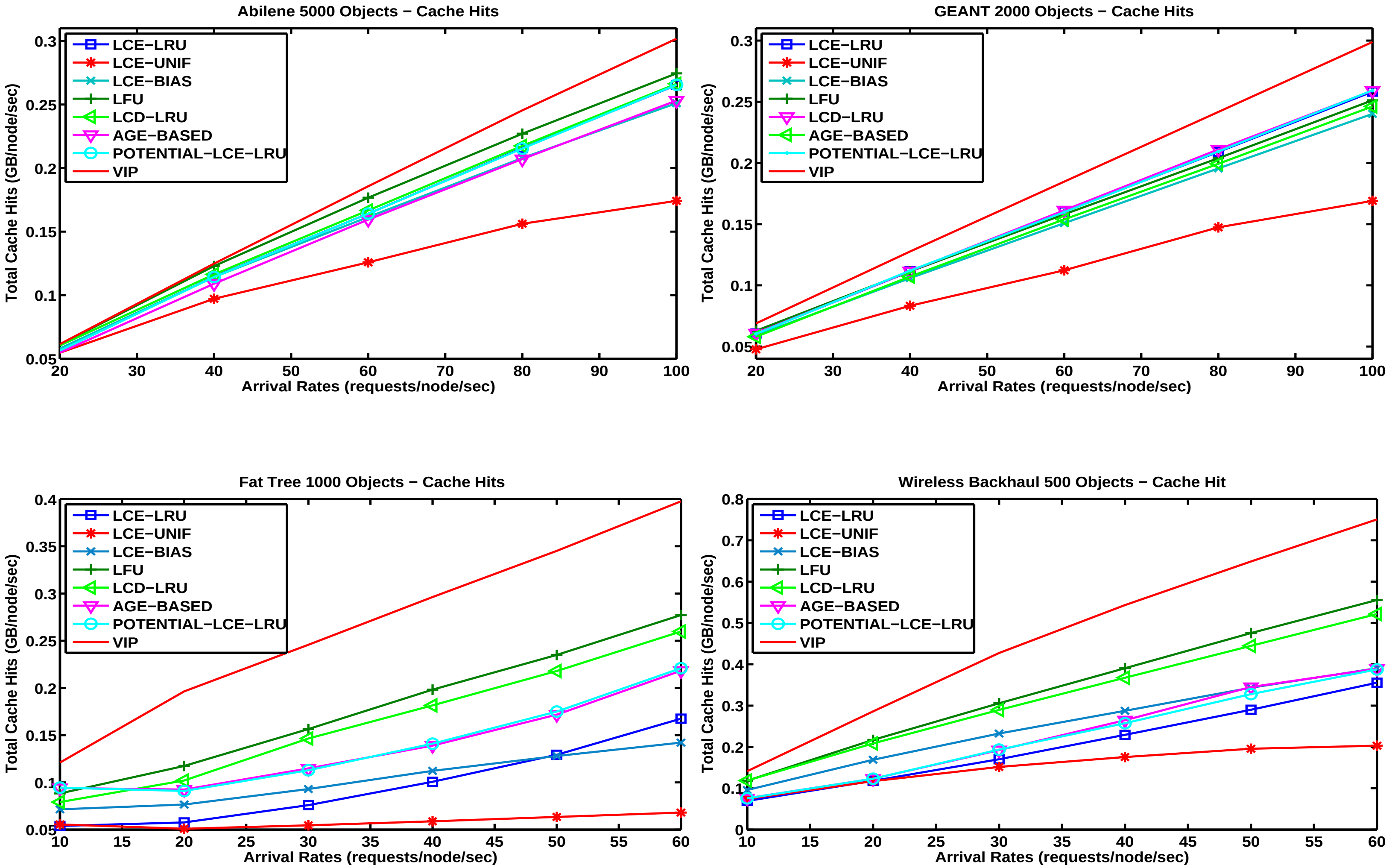
Numerical Experiments: Caching and Forwarding

- Arrival Process: IPs arrive according to Poisson process with same rate.
- Content popularity follows Zipf (0.75).
- Interest Packet size = 125B; Chunk size = 50KB; Object size = 5MB.
- Baselines:
  Caching Decision: LCE/LCD/LFU/AGE-BASED
  Caching Replacement: LRU/BIAS/UNIF/LFU/AGE-BASED
  Forwarding: Shortest path and Potential-Based Forwarding

Numerical Experiments: Delay Performance



Numerical Experiments: Cache Hit Performance



Numerical Experiments: Congestion Control

- α-fair utility functions with $\alpha = 1$ (proportionally fair), $\alpha = 2$, $\alpha \to \infty$ (max-min fair).

- Utility-delay comparison of Stable Caching VIP Algorithm with Congestion Control with AIMD Window-base congestion control with PIT-based forwarding and LRU caching (Carofiglio et al. 2013).

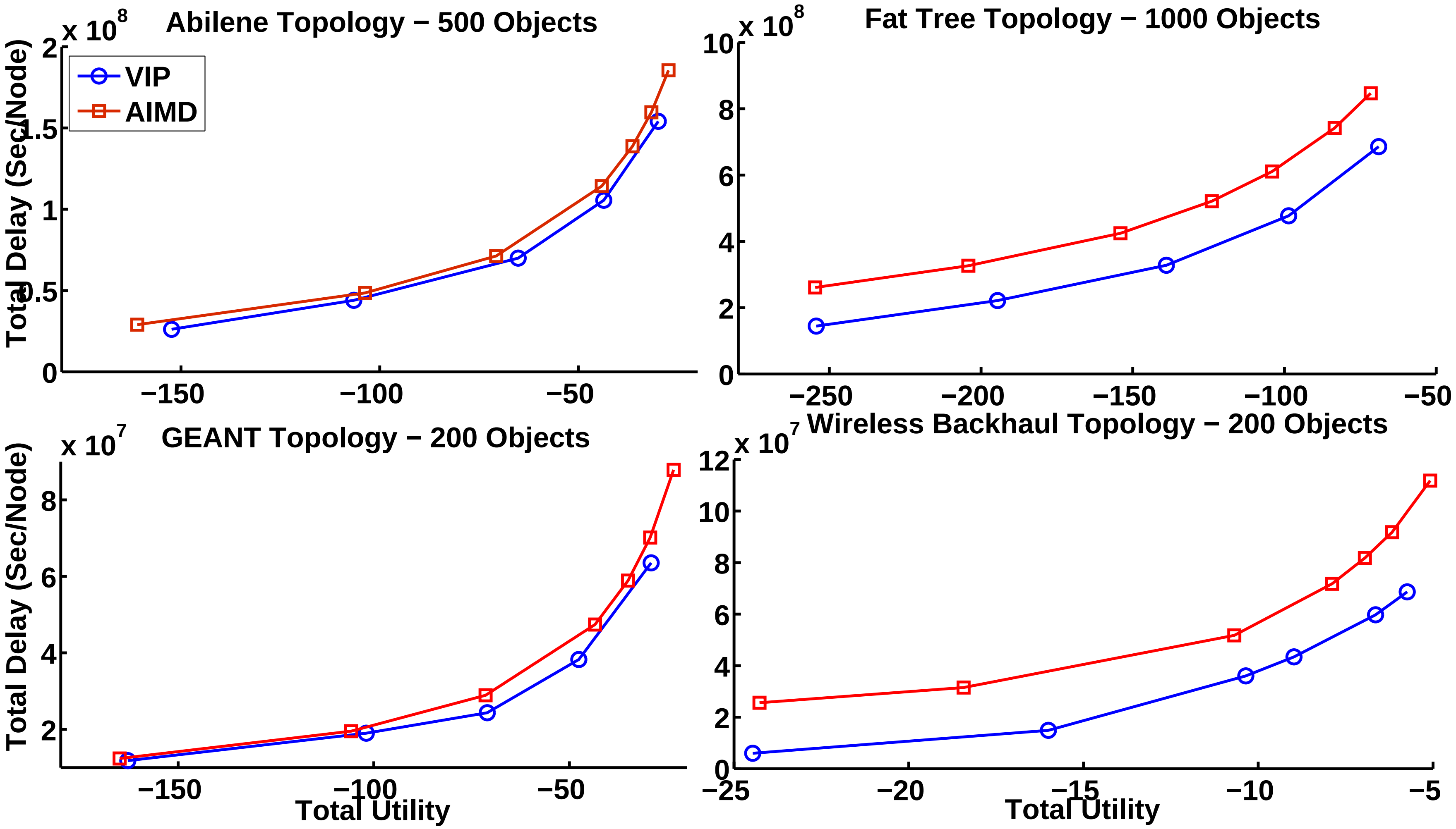
Network Parameters

- Abilene: 500 objects, cache size 500 MB (100 objects), link capacity 500 Mb/s; all nodes generate requests and can be data sources.

- Fat Tree: 1000 objects, cache size $1GB$ (200 objects); CONSUMER nodes generate requests, REPOs are source nodes.

- Wireless Backhaul: 200 objects, cache size $100MB$ (20 objects), link capacity 500 Mb/s; CONSUMER nodes generate requests, REPO is source node.

Numerical Experiments: Comparison with AIMD



Conclusions

- General VIP framework for caching, forwarding and congestion control.
- Distributed caching, forwarding, congestion control algorithms which maximize aggregate utility subject to network layer stability.
- Content-centric congestion control enables fairness among content types.
- Experimental results: superior performance in user delay, rate of cache hits, utility-delay tradeoff.
- VIP algorithms have flexible implementation wrt to caching, forwarding, congestion control.