Trade & Cap: A Customer-Managed, Market-Based System for Trading Bandwidth Allowances at a Shared Link

Azer Bestavros

Computer Science Department
Boston University

Joint work with

Jorge Londono (BU→U Pontificia Bolivariana)
Nikos Laoutaris (BU→Telefonica)

http://www.cs.bu.edu/groups/wing

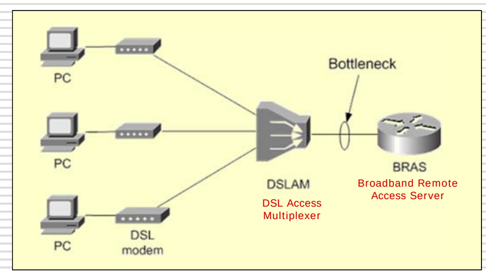
Usenix/ACM NetEcon'10, Vancouver, Canada
October 3, 2010

Today's last mile

The perils of the fixed pricing model

- It's here to stay; metered pricing rejected
- Implications:
  - Customer has no incentive to save bandwidth
  - ISP cost depends on peak demand – 95/5 rule
  - Reigning in bandwidth hogs is incompatible with Net Neutrality
- Must devise mechanisms that take ISPs out of the "traffic shaping" business

DSLAM "last-mile" architecture



Traffic shaping done at BRAS

Solution: Create a marketplace

- Recognize the two types of user traffic:
  - Reserved Traffic (RT)
    - For interactive browsing, VoIP, messaging, gaming, …
    - Limited bandwidth; highly sensitive to response time
  - Fluid Traffic (FT)
    - P2P, Network backup, Netflix/software downloads, …
    - Open-ended bandwidth; less sensitive to response time
- Create a marketplace:
  1. Give users rights to DSLAM bandwidth, and
  2. Let users trade RT/FT allocations over time

The Marketplace

- Each user gets a fixed budget per epoch
  - Budget proportional to level of service
  - An epoch is a fixed number of time-slots, e.g., 1 day = 288 5-min slots
- Trade & Cap
  - User engages in a pure strategies game that yields a schedule for its RT bandwidth
  - User acquires as much FT bandwidth as its remaining budget would allow

Trading Phase: Strategy Space

- Session:

  An RT session is the sequence of slots during which an RT application is active
- Slack:

  User may have flexibility in scheduling RT sessions; slack specifies the number of slots that an RT session is allowed to be shifted back/forth
- Strategy Space:

  The set of all possible arrangements of RT sessions within allowable slack define the strategy space for a user

Trading Phase: Cost Function

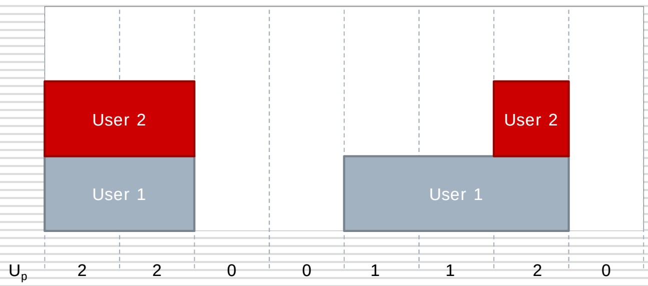
- Let x_{ik} be the bandwidth used in slot k by a chosen RT session schedule for user i.
- The cost incurred by user i is given by:

$$c_i = \sum_{k \in slots} x_{ik} \cdot U_k = \sum_{k \in slots} x_{ik} \left(\sum_{j \in users} x_{jk} \right)$$

- Cost of user i depends on the choices made by other users – hence the game!

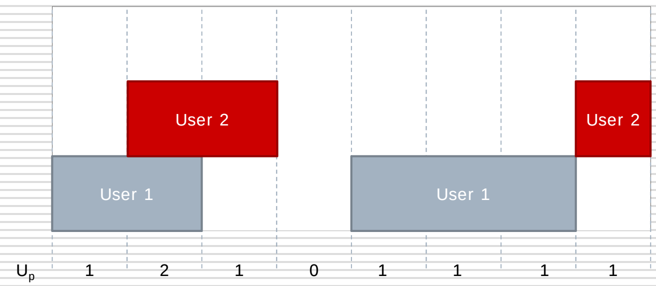
Trading Phase: Illustration

Cost(User 2) = 6



Trading Phase: Illustration

Cost(User 2) = 4



Trading Phase: Best Response

- BR of user i is a schedule of RT sessions that minimizes its cost c_i
- Computing BR is NP-hard, equivalent to solving a generalized knapsack problem
- Dynamic programming solution is pseudo-polynomial in the product of the number of sessions and number of slots
- Scales well for all practical settings – 100s of users and 100s of slots

Trading Phase: Findings

- Provably converges to Nash Equilibrium, even in presence of constraints
- For n users, Price of Anarchy is n, but in practice below 2, especially for n>10
- Experimentally, large reduction of peak utilization, even with small flexibility

Capping Phase: Best Response

- BR of user i is to maximize total FT allocation

$$w_i = \sum_{k \in slots} w_{ik}$$

subject to the budget constraint

$$\sum_{k \in slots} w_{ik} \cdot \left(U_0 + \sum_{j \in users} w_{jk} \right) = B_i - c_i$$

Capping Phase: Budget

- Let V be some desirable upper bound on the total traffic per slot
- The ISP sets a target capacity $C = V/R$, where $R \geq 1$ reflects its "resistance" to traffic
- The ISP allocates C in some proportion (e.g., equally) to all n users over all slots
- This constitutes the budget B assigned to a user over an epoch T

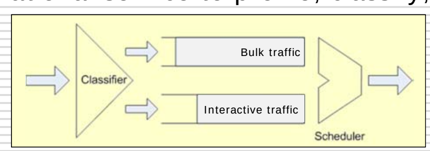
$$B = \frac{C}{n} \cdot T$$

Capping Phase: Findings

- Locally computing BR is efficient using Lagrange Multipliers method
- Provably, converges to a unique global (social) optimum that maximizes the FT allocations of all users (thus could be done centrally by ISP)
- Experimentally, smoothes the aggregate RT+FT traffic to any desirable level controlled by the resistance parameter R

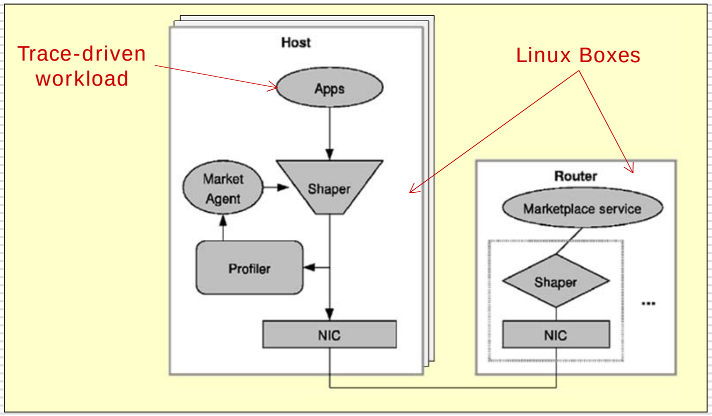
Trade & Cap: Implementation

- On Client Side (e.g., DSL Modem):
  - + Strategic agent to execute Trade & Cap
  - + Operational service to profile, classify, and shape



- ISP Side (e.g., DSLAM or BRAS):
  - + Support exchange between strategic agents
  - + Enforce total traffic/slot/user from Trade & Cap

Trade & Cap: Implementation



Trade & Cap: Implementation notes

- User Input:
  - As simple as checking box to join marketplace, and as elaborate as micromanaging RT slacks
  - May set a fraction of "budget" as insurance
- Client-side Profiler:
  - May be explicitly controlled by applications (or user settings)
- Client-side Traffic Shaper:
  - Work-conserving (not reservation based) Linux Hierarchical Token Bucket (HTB)
  - Allows FT to use underutilized RT bandwidth

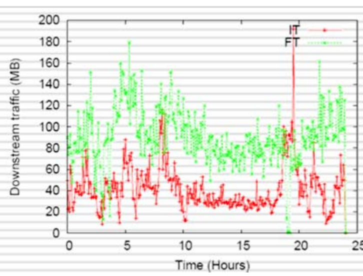
Experimental Evaluation

Workload

Derived from WAN traces of MAWI project†

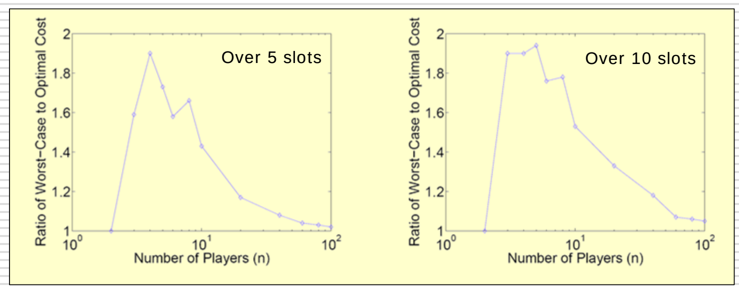
- Identify users from volume and direction of flows to known ports (e.g., most traffic destined to port 80)
- Identify user RT sessions using thresholds on per-IP traffic intensities over time
- Slack introduced using various models (e.g., fixed, proportional, etc.)

Period	2009-03-31 00:00 – 2009-03-31 23:59
Total packets	1,551,089,845
TCP packets	1,194,409,653
UDP packets	4,321,852
Total TCP bytes (payload)	924,540,189,060



†Reported results are negatively impacted by less-than-ideal (atypical) trace.

Trading Phase: Experimental PoA

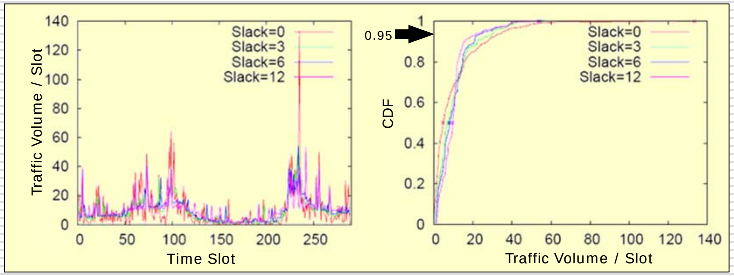


Theoretical PoA is n but not in practice

Trading Phase: Smoothing effect

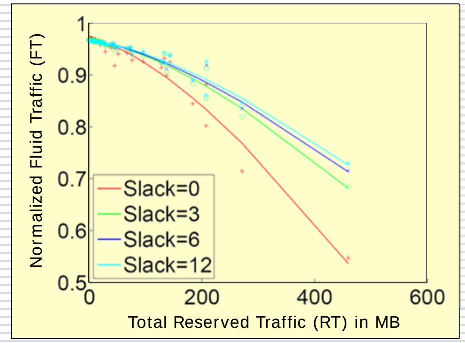
Value proposition to ISPs

Max Slack	Reduction in 95%
3	15%
6	24%
12	31%



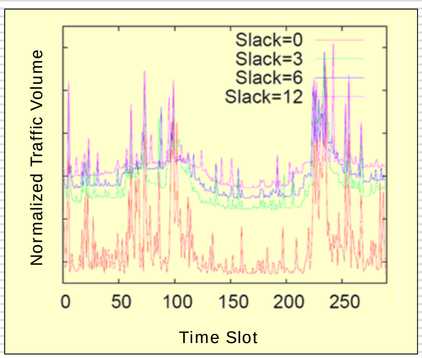
Trade & Cap: Flexibility pays off!

Value proposition to customers



Trade & Cap

A win-win for ISPs and customers



Trade & Cap: Beyond DSLAMs

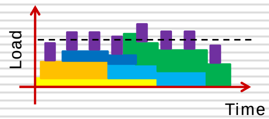
- Trade & Cap is a *general* mechanism
  It can be used to coordinate how a shared resource is used by selfish parties who are not subject to the "pay as you go" model – e.g., "fixed pricing"
- Examples
  - Coordinating consumption of "reserved" versus "fluid" (CPU/network) capacities of VMs sharing a single host
  - Coordinating "reserved" versus "fluid" bandwidth utilization by multiple ISP customers (e.g., enterprises)

Selfish Resource Packing Problems

Shared bandwidth arbitration

- Trade & Cap

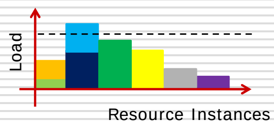
  A temporal packing game



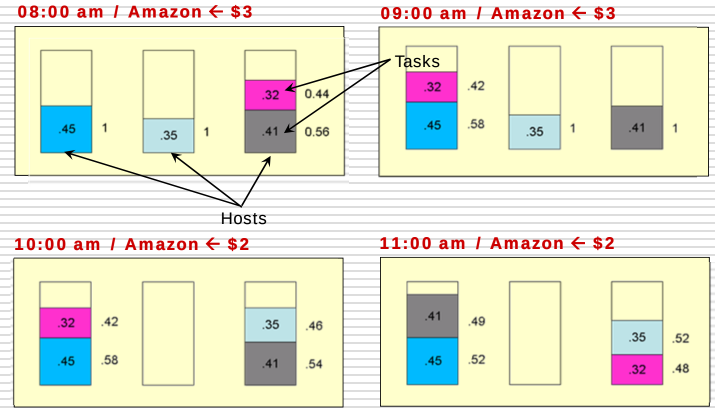
Cloud resource acquisition

- Colocation Games

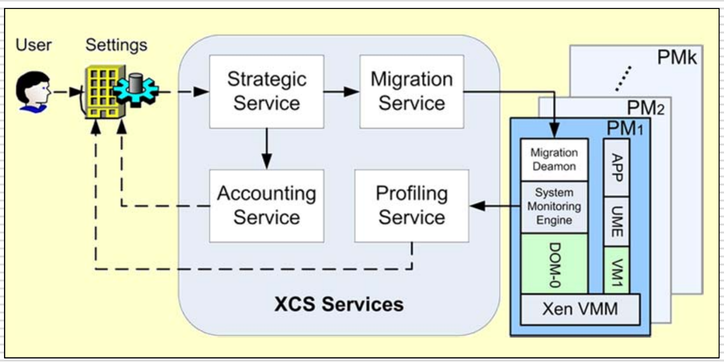
  A spatial packing game



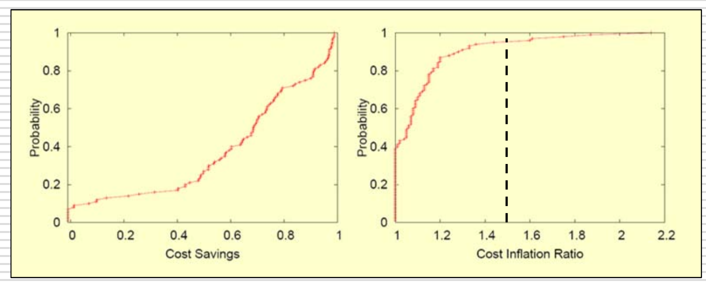
Colocation Games



CloudCommons: Architecture



CloudCommons: Benefit to users



Multi-dimensional Planet-Lab trace-driven experiments
(Overheads/costs of all XCS services included)

Conclusion

- In many settings, resource management can only be seen as a strategic game among rational peers
- By setting up the right mechanism, one can ensure convergence and efficiency
- New services are needed to support strategic and operational aspects of these mechanisms

➔ Trade & Cap is an example of such mechanisms
  - It coordinates the shared use of a resource by trading in "rights to quality" for "volume"
  - It has been implemented in a last-mile setting as a proof of concept with very promising performance

Publications

"netEmbed: A service for embedding distributed applications (Demo)". Londono and Bestavros. *ACM/Usenix Middleware'07*.

"netEmbed: A resource mapping service for distributed applications". Londono and Bestavros. *IEEE/ACM IPDPS'08*.

"Colocation games with application to distributed resource management". Londono, Bestavros, and Teng. *USENIX HotCloud'09*.

"*Colocation as a Service: Strategic & operational cloud colocation services*". Ishakian, Sweha, Londono, and Bestavros. *IEEE NCA'10*.

"Trade & Cap: A customer-managed system for trading bandwidth at a shared link". Londono, Bestavros, and Laoutaris. *ACM/Usenix NetEcon'10*.

http://csr.bu.edu/cc