An exact penalty function view of resource allocation and congestion control in the Internet

Srisankar Kunniyur

University of Pennsylvania

http://www.seas.upenn.edu/∼kunniyur

R. Srikant

University of Illinois at Urbana-Champaign

http://comm.csl.uiuc.edu/∼srikant

March 18, 2002

Overview

- Resource allocation and congestion control (Kelly et al.)
- Penalty function approach to obtain decentralized congestion controllers (Kelly etal.)
- Dual approach (Low et al.)
- Shadow prices via penalty function approach
- Interpretation: Low-loss, low-delay network operation with high utilization (AVQ)
- Decentralized design of AVQ parameters

System Model

- $\mathcal{L}$ - Set of links l with capacity C_l
- γ_l - Desired utilization at link l
- $\gamma_l C_l$ - Target capacity at link l
- User r - Subset of $\mathcal{L}$
- $\mathcal{R}$ - Set of all users
- S - Routing matrix, i.e., $S_{rl} = 1$ if $l \in r$, otherwise $S_{rl} = 0$.
- x_r - Rate of User r
- $U_r(x_r)$ - Utility of rate x_r to User r
- Let $x = (x_r, r \in \mathcal{R})$ and $C_\gamma = (\gamma_l C_l, l \in \mathcal{L})$

System Problem ... (Kelly et al.)

$$\boxed{\text{SYSTEM}(U, S, \gamma)}$$

$$\max_{x} \sum_{r} U_r(x_r)$$

subject to

$$\begin{aligned} S^T x &\leq C_\gamma \\ x &\geq 0 \end{aligned}$$

Maximize aggregate utility subject to capacity and non-negativity constraints

Penalty Function Formulation (Kelly et al.)

- Penalty function approach to the system problem:

$$\max_{\{x_r\}} \sum_r U_r(x_r) - \beta \sum_{l \in L} \int_0^{\sum_{i:l \in i} x_i} p_l(z)\,dz$$

- $\beta p_l(x)$ is the penalty for exceeding the capacity at link l
- Can achieve the solution to the above problem in a decentralized manner using congestion-controllers (steepest ascent algorithm):

$$\frac{dx_r}{dt} = 1 - \beta (U_r'(x_r))^{-1} \sum_{l:l \in r} p_l\Big(\sum_{j:l \in j} x_j \Big)$$

Can we solve SYSTEM(U, S, γ) exactly?

- By appropriate choice of penalty functions
- Penalty is finite
- Called Exact Penalty Functions

Exact Penalty Functions

- Rewrite the original penalty function:

$$\max_{\{x_r\}} \sum_r U_r(x_r) - \beta \sum_{l \in L} \int_0^{\sum_{i:l \in i} x_i} p_l(z, \tilde{C}_l)\, dz$$

- Parametrize the penalty functions using a parameter called the virtual capacity
- $\tilde{C}_l$ can be thought of as a parameter that determines when congestion feedback is provided

Goal: Design the parameters $\{\tilde{C}_l\}$ such that the penalty function formulation solves SYSTEM(U, S, γ)

How to estimate $\{\tilde{C}_l\}$ in a heterogeneous network?

- Update $\tilde{C}_l$ at each link as follows:

$$\dot{\tilde{C}}_l = \alpha_l(\gamma_l C_l - \lambda_l)$$

where λ_l is the total flow into the link

- When **total arrival rate $<$ target arrival rate**:
increase $\tilde{C} \Rightarrow$ reduce number of packets marked $\Rightarrow$ total arrival rate increases

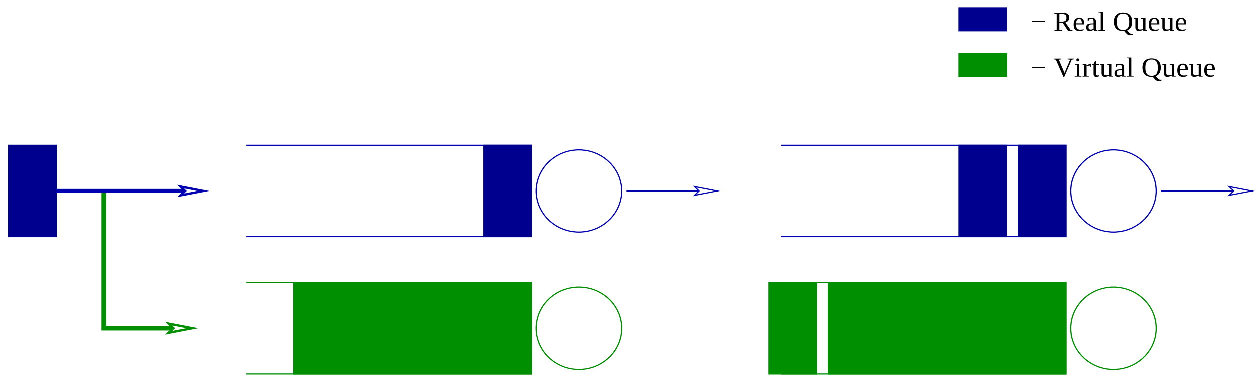
- When **total arrival rate $>$ target arrival rate**:
decrease $\tilde{C} \Rightarrow$ increase number of packets marked $\Rightarrow$ total arrival rate decreases

- At the equilibrium point, **total arrival rate = target arrival rate**

- α determines speed of adaptation

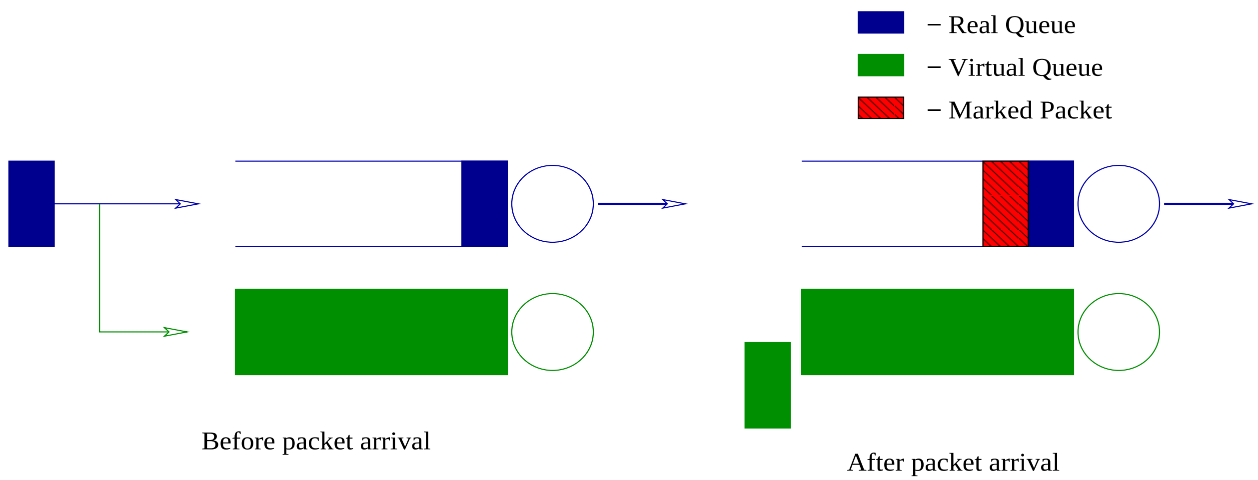
AVQ Algorithm

- Consider a link with capacity C
- Assume desired utilization is $\gamma < 1$
- Maintain a virtual queue with capacity $\tilde{C}$ (*virtual*-*capacity*) and buffer size B
- Upon each arrival, add a fictitious packet to the VQ
- If VQ overflows, discard the fictitious packet and mark/drop a packet in the real queue
- Update $\tilde{C}$ periodically by measuring the total average arrival rate



Before packet arrival

After packet arrival


− Real Queue
− Virtual Queue
− Marked Packet
Before packet arrival
After packet arrival

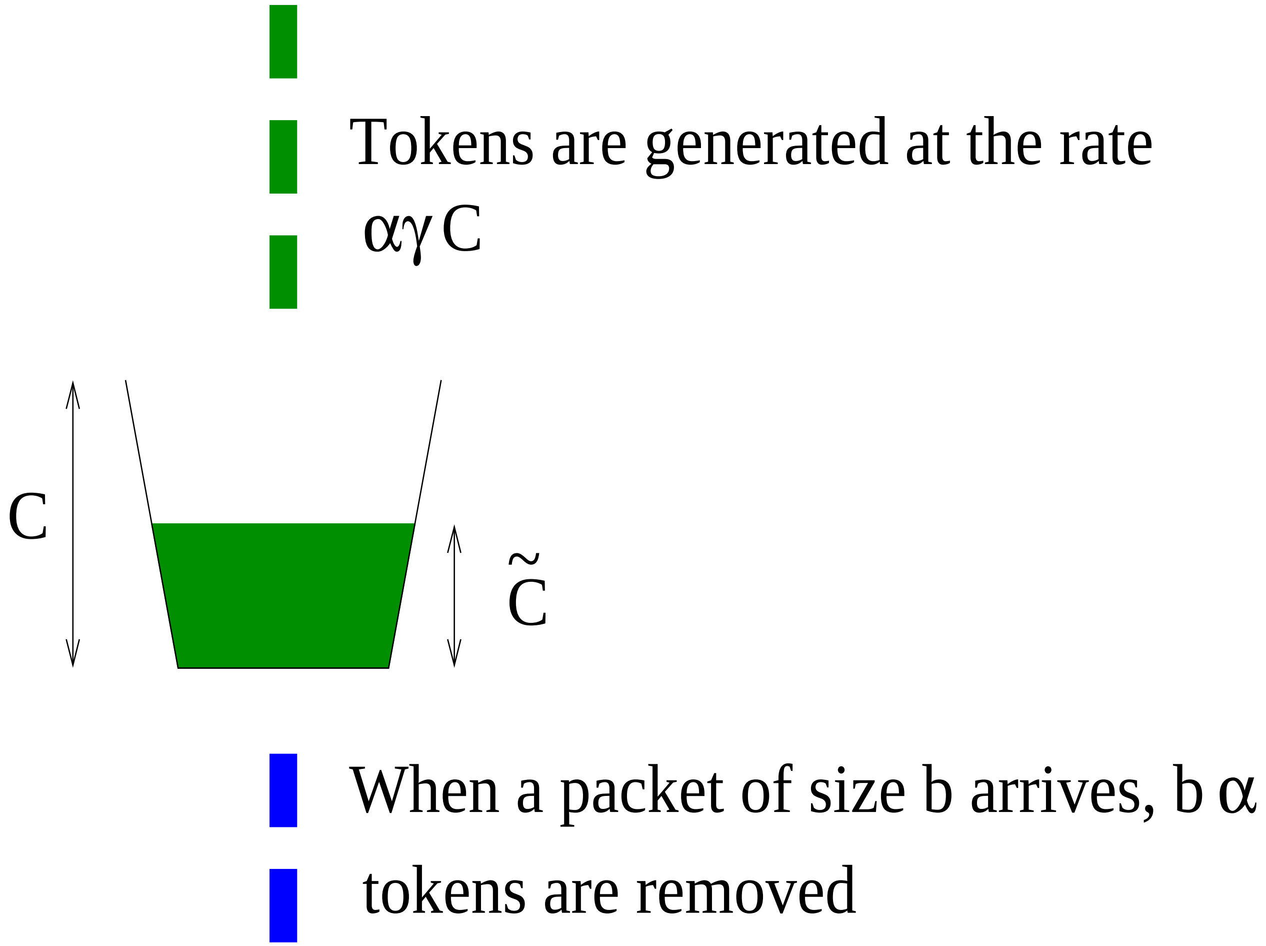


Fig. 1: Token Bucket interpretation

How to choose α to ensure stability?

- System comprises of:
  - Congestion-controllers at the sources
  - AVQ algorithm at the links
- In the absence of feedback delays, system is **semi-globally exponentially stable** for **small** α
  - Proof by Singular Perturbations
  - Adapt at the link at slower rate
- Lagrange multipliers of SYSTEM(U, S, γ) are given by $\{\beta p_l(\gamma_l C_l, \tilde{C}_l^*)\}$
- How to choose α for non-zero round-trip delays?

System Model

- Consider a single link
- Users with fixed round-trip propagation delay d
- Assume users employ propotional congestion controller ($U(x) = \log(x)$)

$$\dot{x}_r = \kappa_r \left(w_r - x_r(t - T_r) p\big(\sum_{j \in R} x_j(t-d), \tilde{C}(t-d)\big) \right)$$

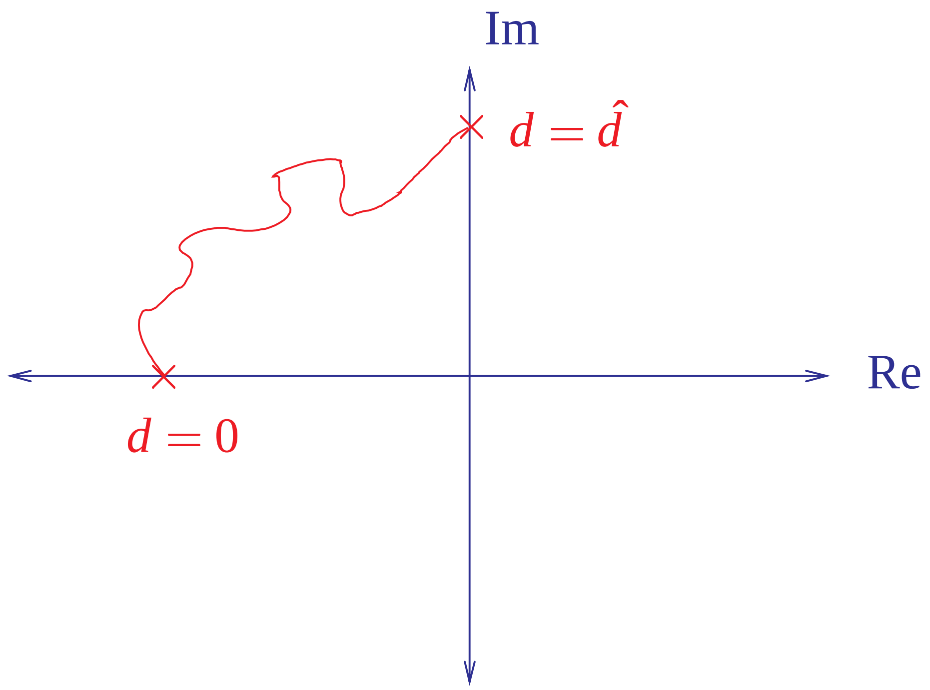
- Update equation at the link is:

$$\dot{\tilde{C}} \quad = \quad \alpha(\gamma C - \sum_{j \in R} x_j(t)) \qquad \forall l \in L$$

Key Idea

- Linearize the system
- Take Laplace transform
- Obtain the characteristic function
- For stability, all roots of the characteristic equation should have negative real parts
- When $d = 0$, the system is stable for all $\alpha > 0$.

Key Idea



$\boxed{\text{System stable for all } d < \hat{d}}$

Key Idea

- Given α, and γ, find the maximum delay $\hat{d}$ such that the system is stable for all $d < \hat{d}$.
- More practical situation is, given d, find α such that the system is stable
- Condition remains the same
- In general, fix any three of the four parameters α, d, γ and N to find a bound on the fourth paarmeter to guarantee stability

Diverse round-trip delays

System Model:

- $d_1(r)$: Delay from the source to the link
- $d_2(r)$: Feedback delay from the link to the source
- $T_r = d_1(r) + d_2(r)$: Total delay
- log utility function

$$\dot{x}_r = \kappa_r \left(w_r - x_r(t - T_r) p\big(\sum_{j \in R} x_j(t - d_1(j) - d_2(r)), \tilde{C}(t - d_2(r))\big) \right),$$

$$\dot{\tilde{C}} = \alpha(\gamma C - \sum_{j \in R} x_j(t - d_j(1))).$$

- (Vinnicombe) In the absence of AVQ, the system of congestion-controllers is stable if:

$$\kappa_r(\hat{p}+\hat{p}_x\gamma C) \leq \frac{\pi}{2T_r} \qquad \forall r \in R \, .$$

- With AVQ, we can show that the system is stable if:

$$\kappa_r(\hat{p}+\hat{p}_x\gamma C) \leq \min\{\frac{1}{4T_r}, \kappa_{\max}\} \qquad \forall r \in R \, ,$$

and

$$\alpha \leq \min\{\frac{\hat{p}}{\sqrt{2}\hat{p}_{\tilde{C}}\gamma C T_{\max}}, \frac{\pi^2 \hat{p}_x \kappa_{\max}}{32\hat{p}_{\tilde{C}} T_{\max}^2}\}.$$

$\boxed{\text{Adapt the virtual capacity at a rate slower than the maximum round-trip delay}}$

Simulations

- Single link with capacity 10 Mbps
- TCP Reno users with average packet length of 1000 bytes
- Propagation delay of each user between 40ms and 130 ms
- Buffer capacity at the link 100 packets
- Study the convergence properties and buffer sizes at the link for the AVQ scheme
- Number of long FTP connections = 180
- Short flows (20 packets each) arrive at the rate of 30 flows per second
- Introduce the short flows at time $t = 100$s

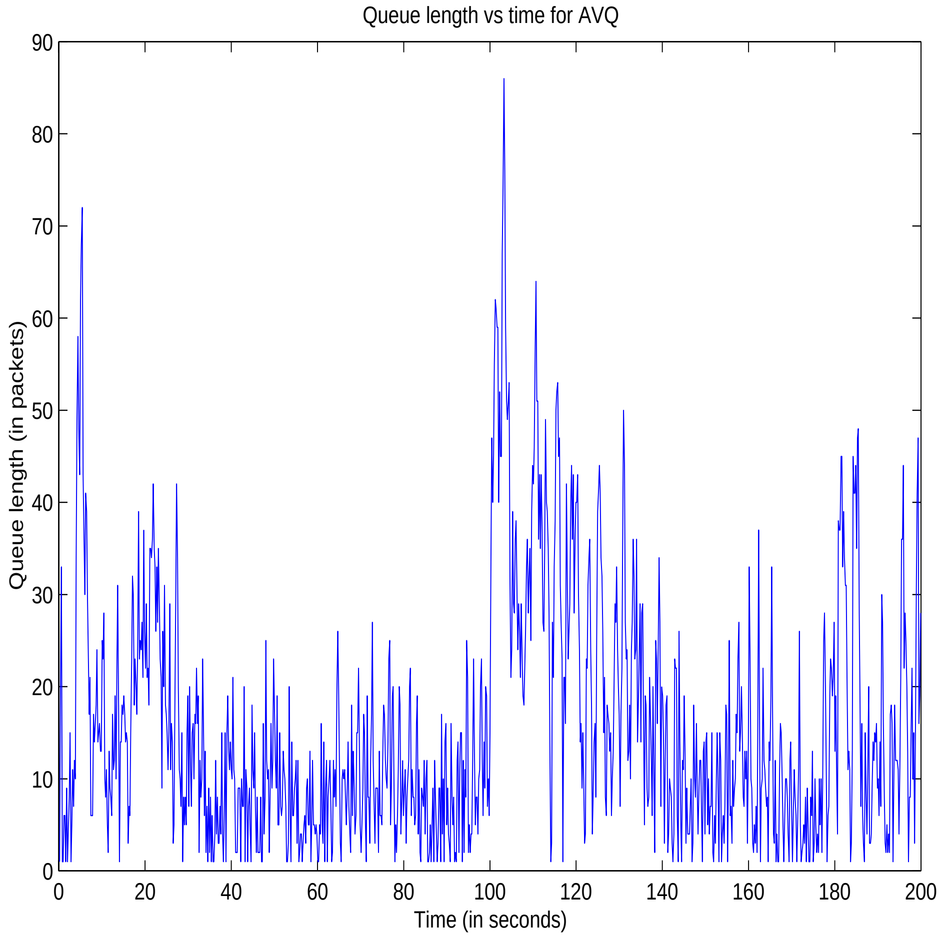


Fig. 2: Queue length vs time for the AVQ scheme

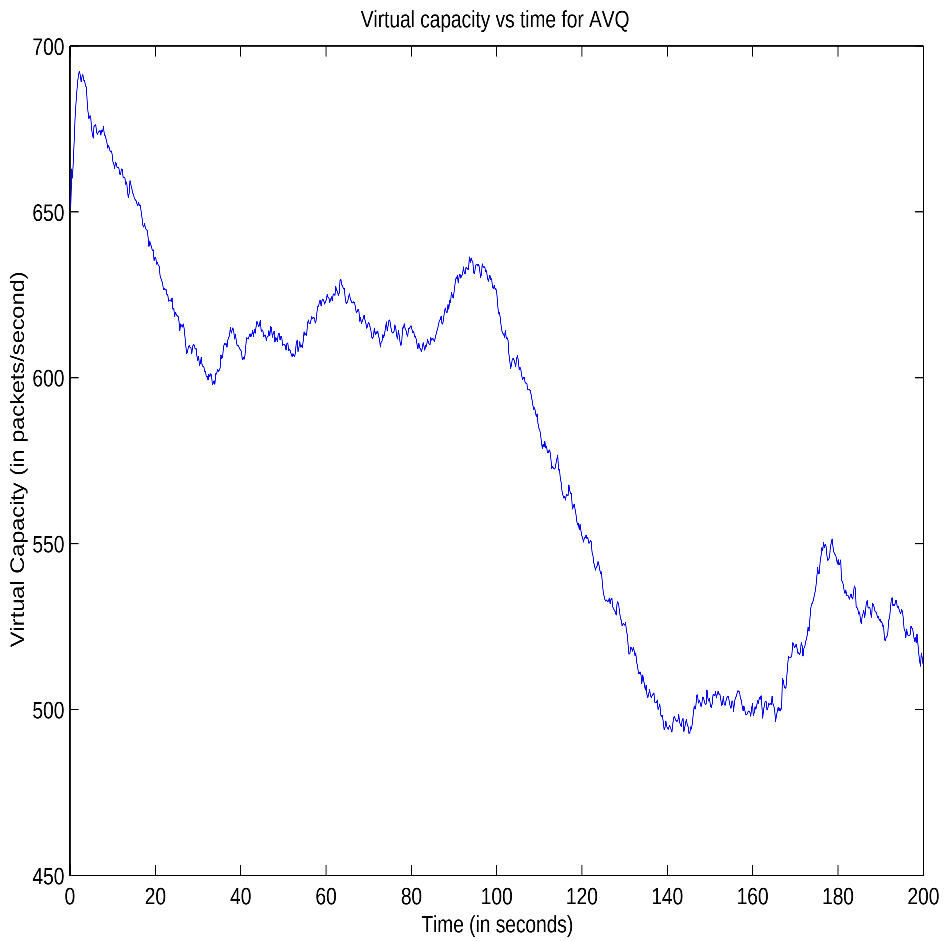


Fig. 3: Virtual capacity vs time for the AVQ scheme

Conclusions

- Presented an easily implementable, robust AQM scheme
- Direct result of using an exact penalty function approach
- Decentralized choice of AQM parameters

All papers can be downloaded from:

http://www.comm.csl.uiuc.edu/∼srikant

or

http://www.seas.upenn.edu/∼kunniyur