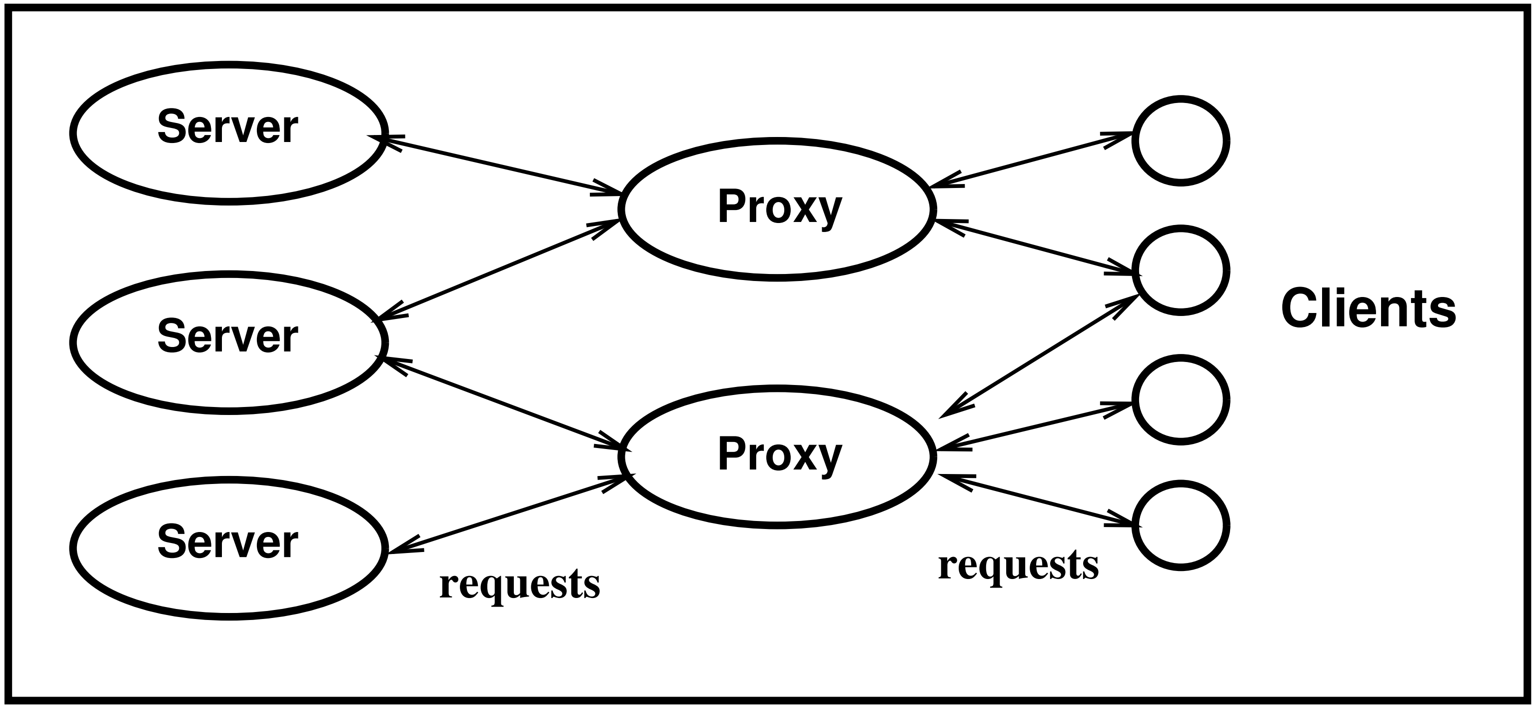
Optimizing Pushes and Pulls in Wide-Area Information Systems

Krithi Ramamritham

Sources, Proxy (Caches) and Clients



proxies can process, integrate, transcode

Pull vs. Push

- **Pull**: Client *pulls* data from source (whenever it suspects that data might have changed at the source).

  + + stateless servers, failures easy to handle

  – – limited scalability

- **Push**: source *push*es data to client (whenever data changes at the source).

▷ pull or push or *both*

▷ how often to pull/push?

The Goal: Keeping Clients "up-to-date"

Given

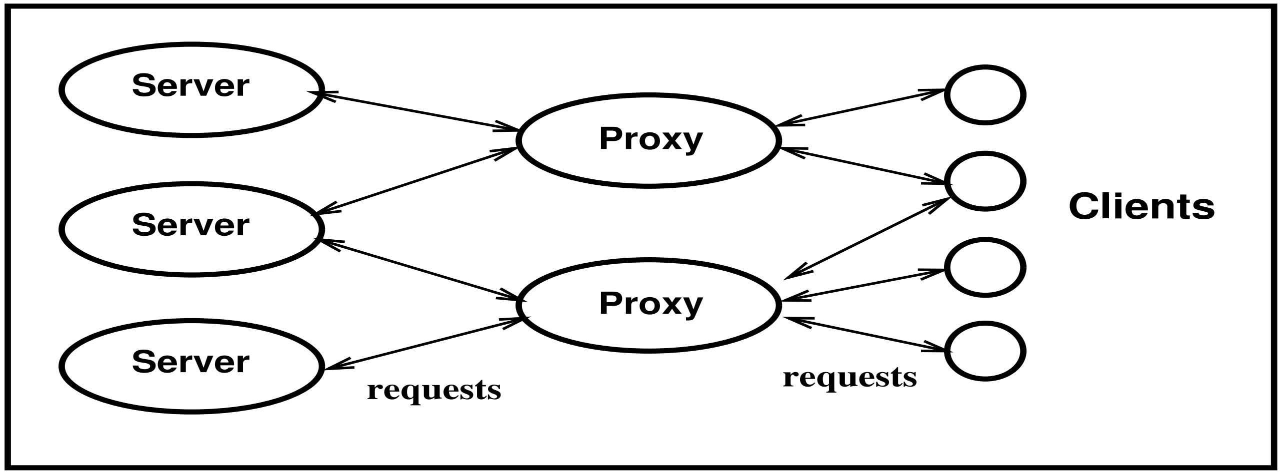
- the dynamics of the data, and
- the clients'/users' need for temporal accuracy
  - ▷ *Company X's stock price should be accurate within 1 rupee.*
  - ▷ *Aircraft Y's position should be no more than 2 mins old.*
  - ▷ *Cricket score should be no more than one over old.*

- *Prompt me if the overall run rate is close to what is needed to clinch a berth in the final.*

Motivating Examples

- *customized sports and news coverage*
  - provide customized *views* of ongoing events
- *continuous monitoring of stocks/auctions*
  - notifications of important events
- *collaborative applications*
  - all clients want to see
    a single consistent (virtual) environment
- *PDA-based applications*
  - optimize data dissemination
    for slow and lossy wireless links

HTTP – pull+pull



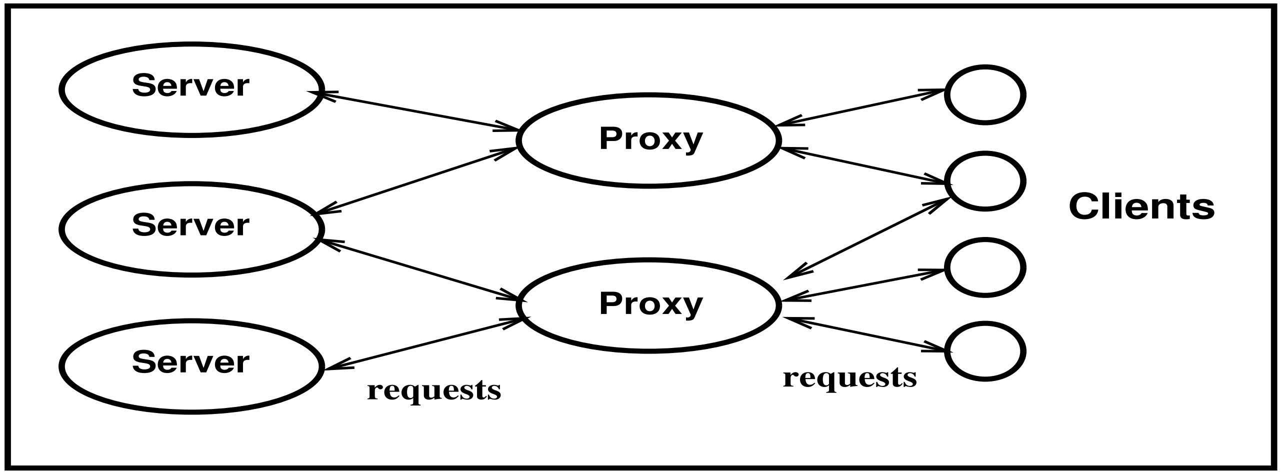
pull pull

Request+Response:

- proxy pulls from server
- client pulls from proxy

▷ how often to pull?

Leases – push+pull



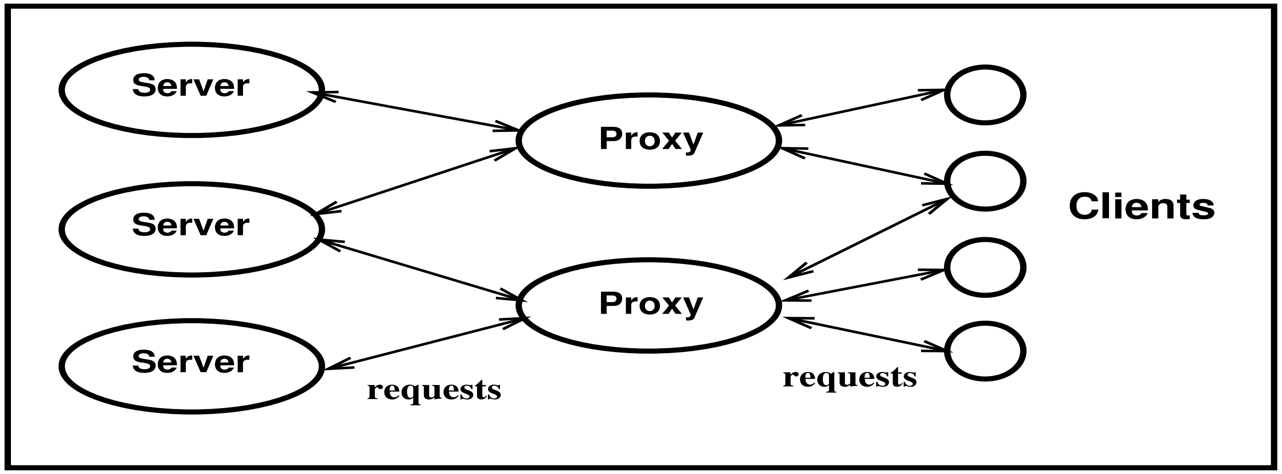
push pull

- server pushes changes to proxy until end of lease
  – client can pull and renew lease
- client pulls from proxy

▷ stateful server with push capability needed

▷ longer lease ⇒ more state info maintained at server

Push-based proxies – pull+push



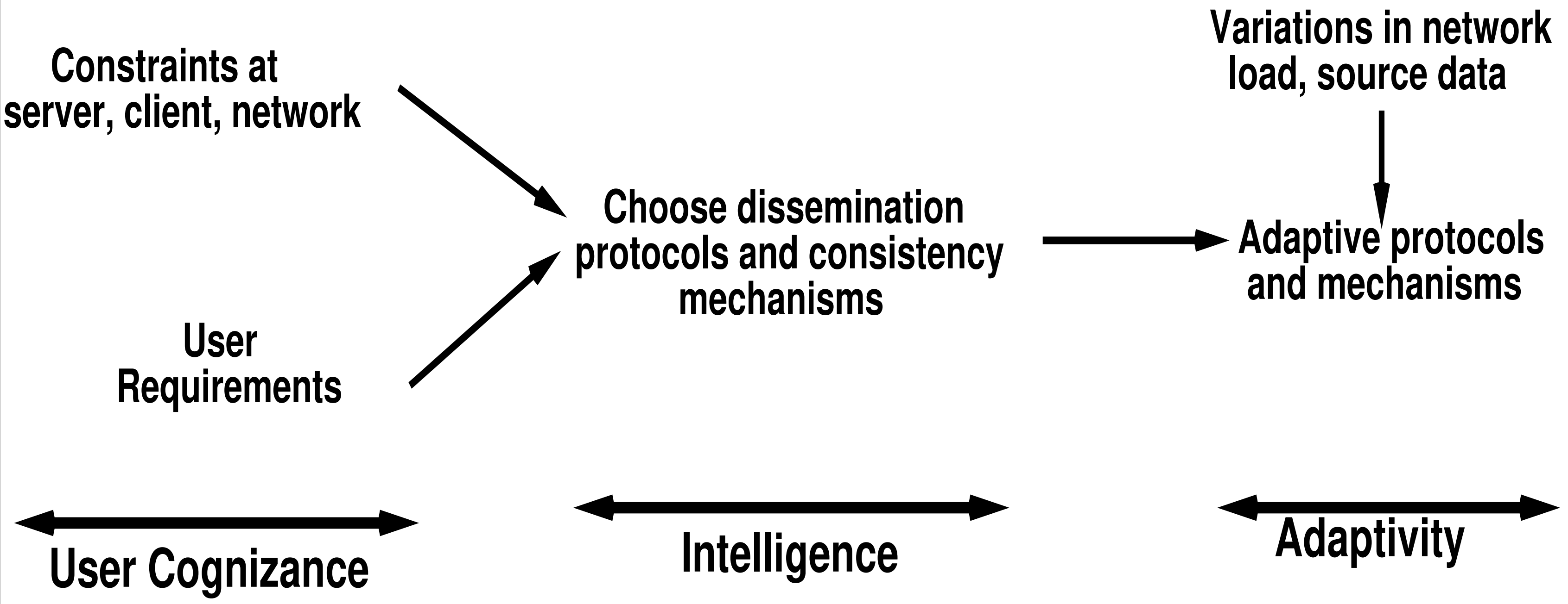
pull push

Adaptable proxies:

- proxy pulls from server
- proxy pushes relevant changes to client

▷ how often to pull?

Optimization Issues



A Specific Problem

Company X's stock price should be accurate within 1 rupee.

- Satisfy this user specified "value constraint" c

$$|state\_at\_Client(t) - state\_at\_Server(t)| < c$$

- in a Web/HTTP context:
  - ▷ proxy pulls from Web server
  - ▷ proxy pushes relevant changes to client

The Web: Time-To-Live (TTL)

- TTL values are attached to cached objects (HTML pages).

  Cached object used until TTL expires

- Source contacted with an `if-modified-since` request.

  Server responds with new page only if the requested object has been modified since the specified time.

▷ *How should TTL values be assigned?*

Goal

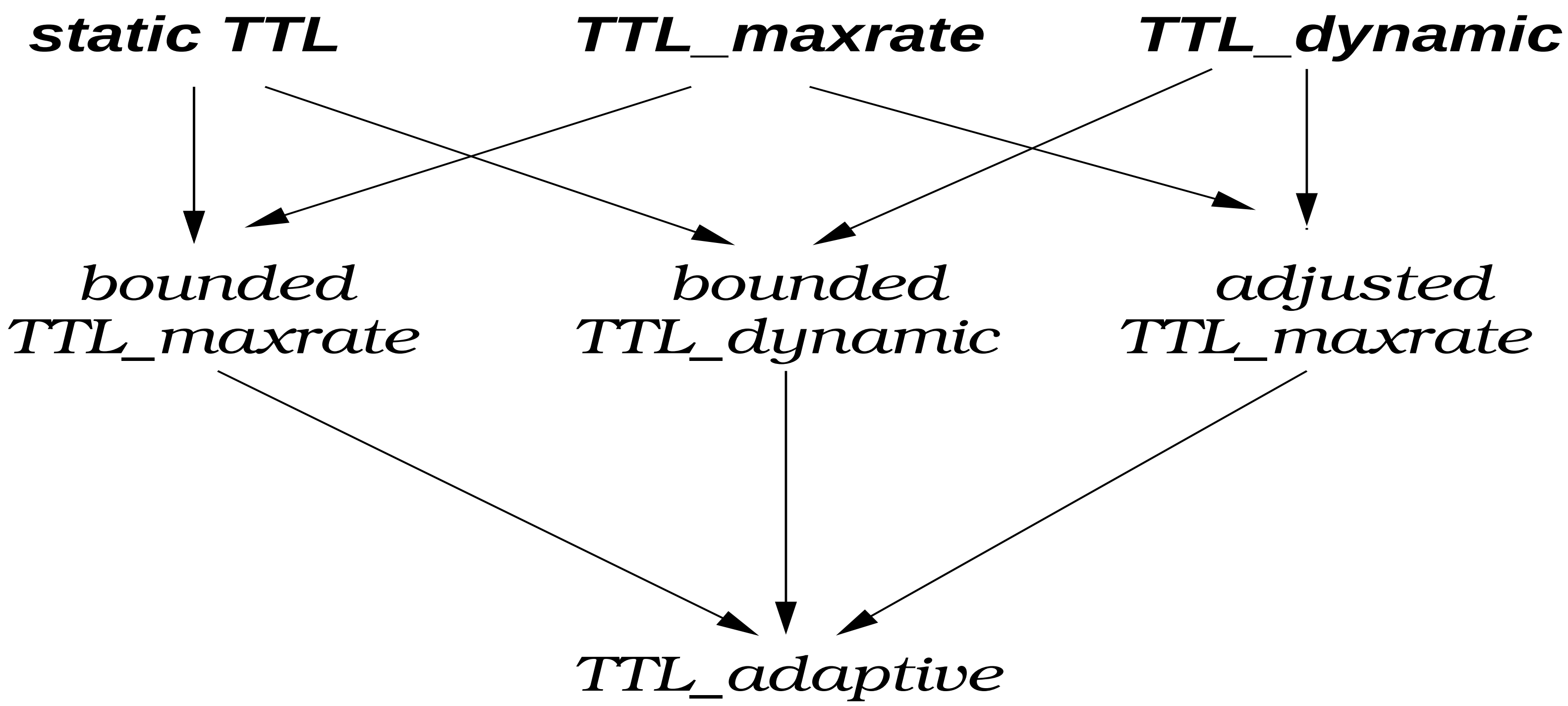
Goal: Minimize

- *#pollings* – the number of times the source is polled
- probability that a user's requirement
  – *to be within c of the source* – is violated

$$VProb = \frac{\sum_{i=1}^{n} t_i}{T}$$

where, during a session of length T, violations occur during intervals of length t_i.

A Suite of Protocols



Adaptive TTL

1. TTL should be within reasonable limits.

   – statically specified $\{TTL\_min, TTL\_max\}$

2. TTL should reflect recent changes

   – $TTL\_dynamic$ reflects recent TTLs and δs

3. Should be able to react to possible quick changes (as observed in the past)

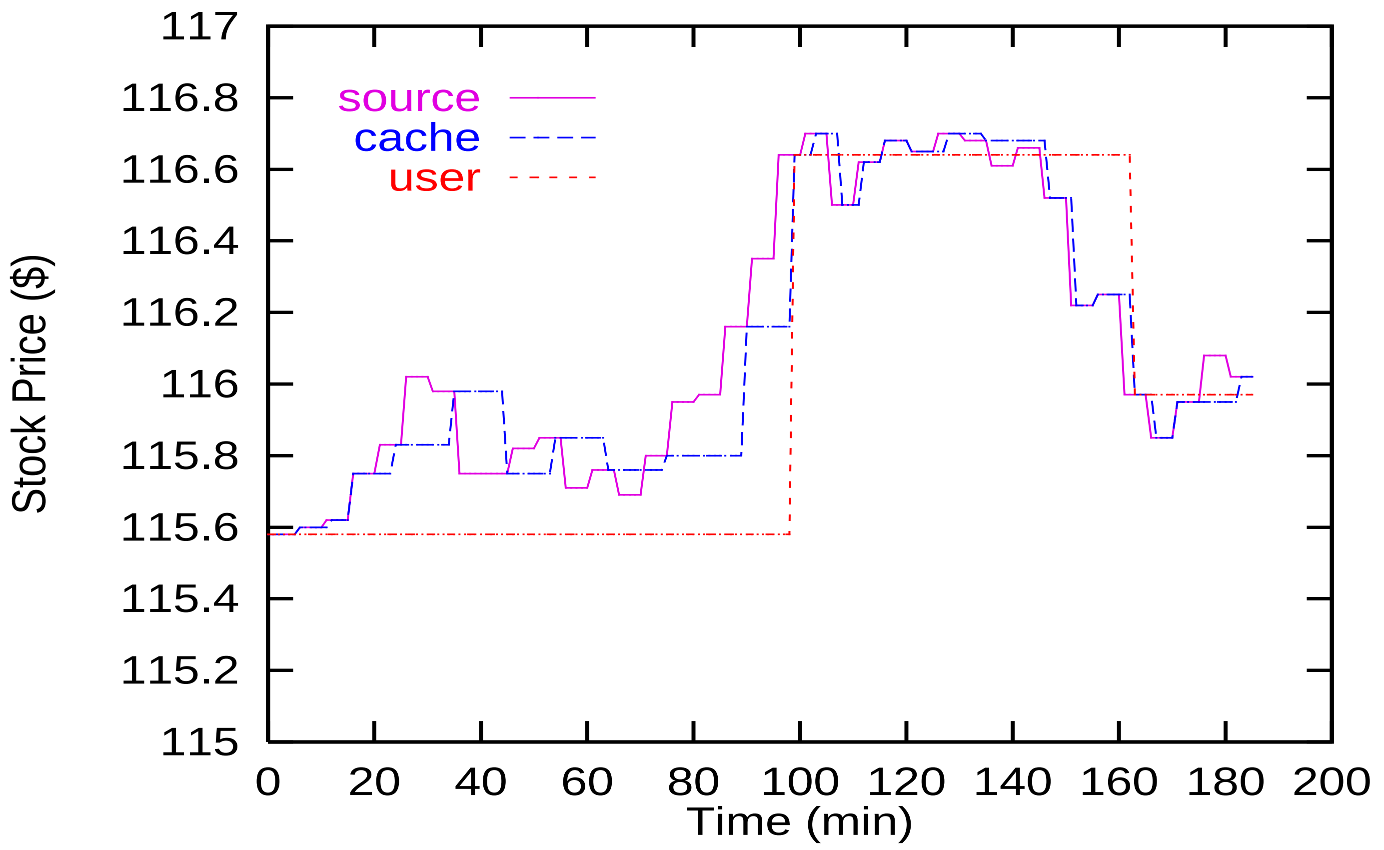
   – $TTL\_maxrate_a$ takes this into account

Adaptive TTL

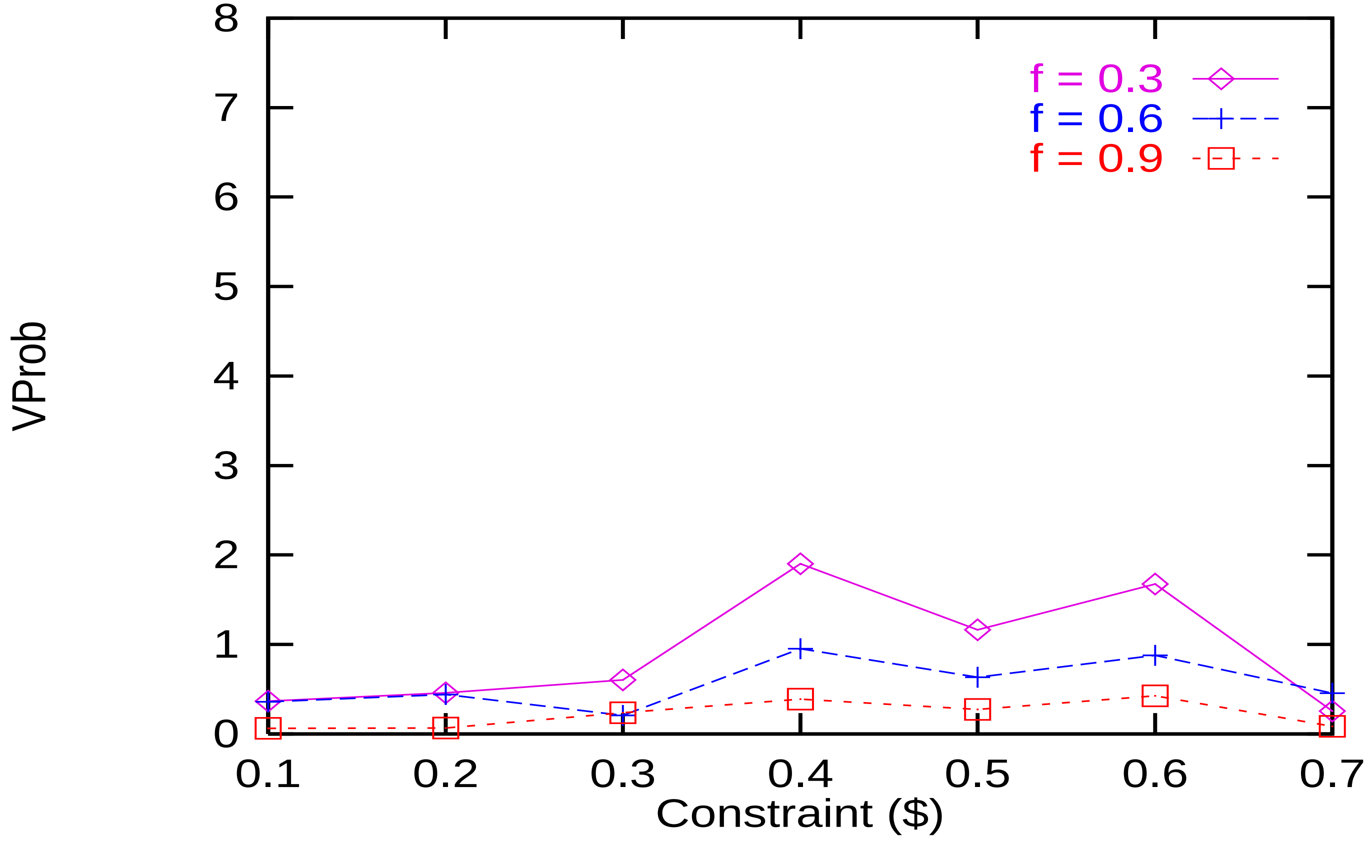
$$
\begin{aligned}
TTL_{adaptive} = Max(&TTL\_min,\\
&Min(TTL\_max,\\
&\quad TTL\_dynamic,\\
&\quad TTL\_maxrate_a))
\end{aligned}
$$

- TTL_min, TTL_max – statically determined bounds
- $TTL\_dynamic$ – TTL based on recent changes
- $TTL\_maxrate_a$ – accomodates previously observed maximum rate of change & the recent change

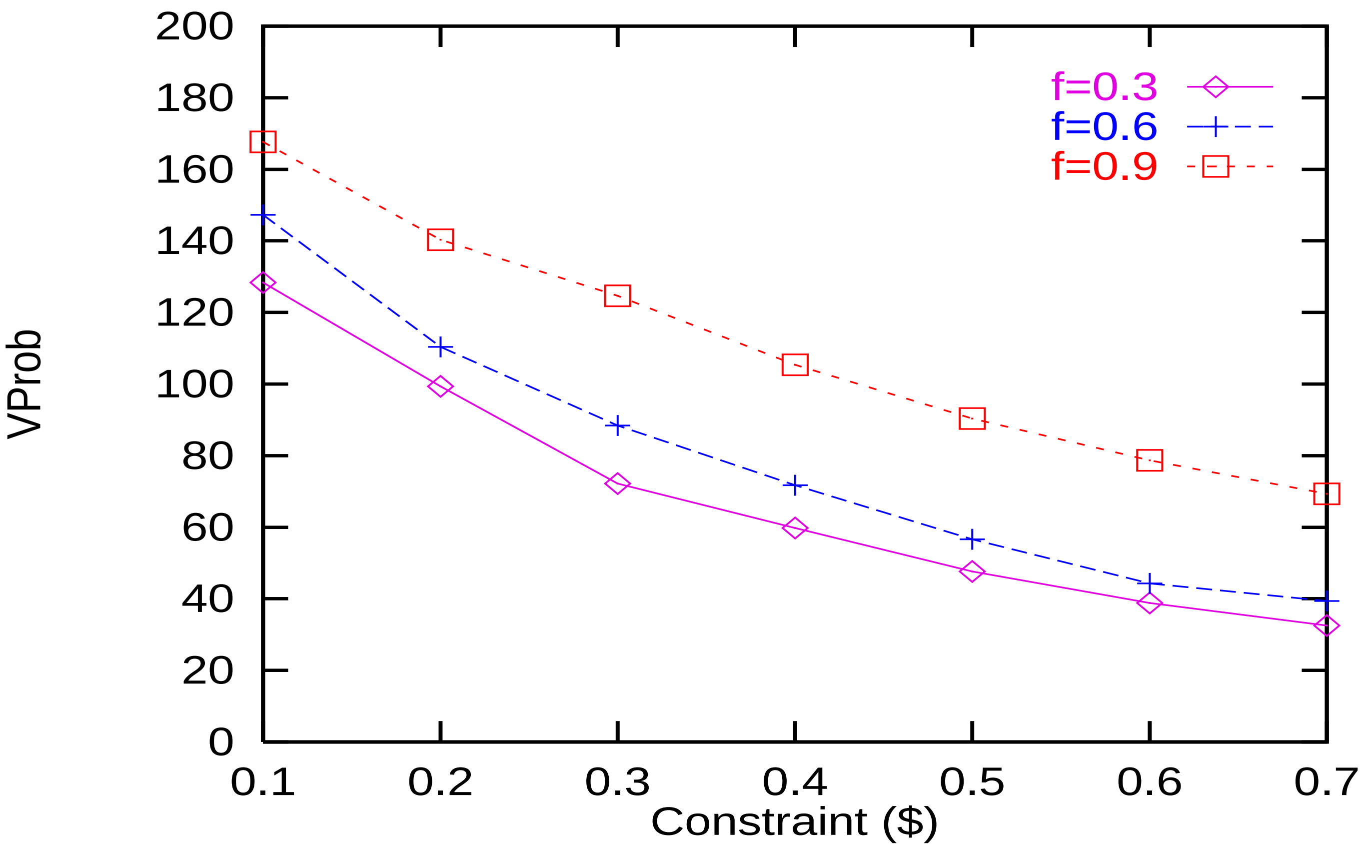
Adaptive TTL – dealing with changes



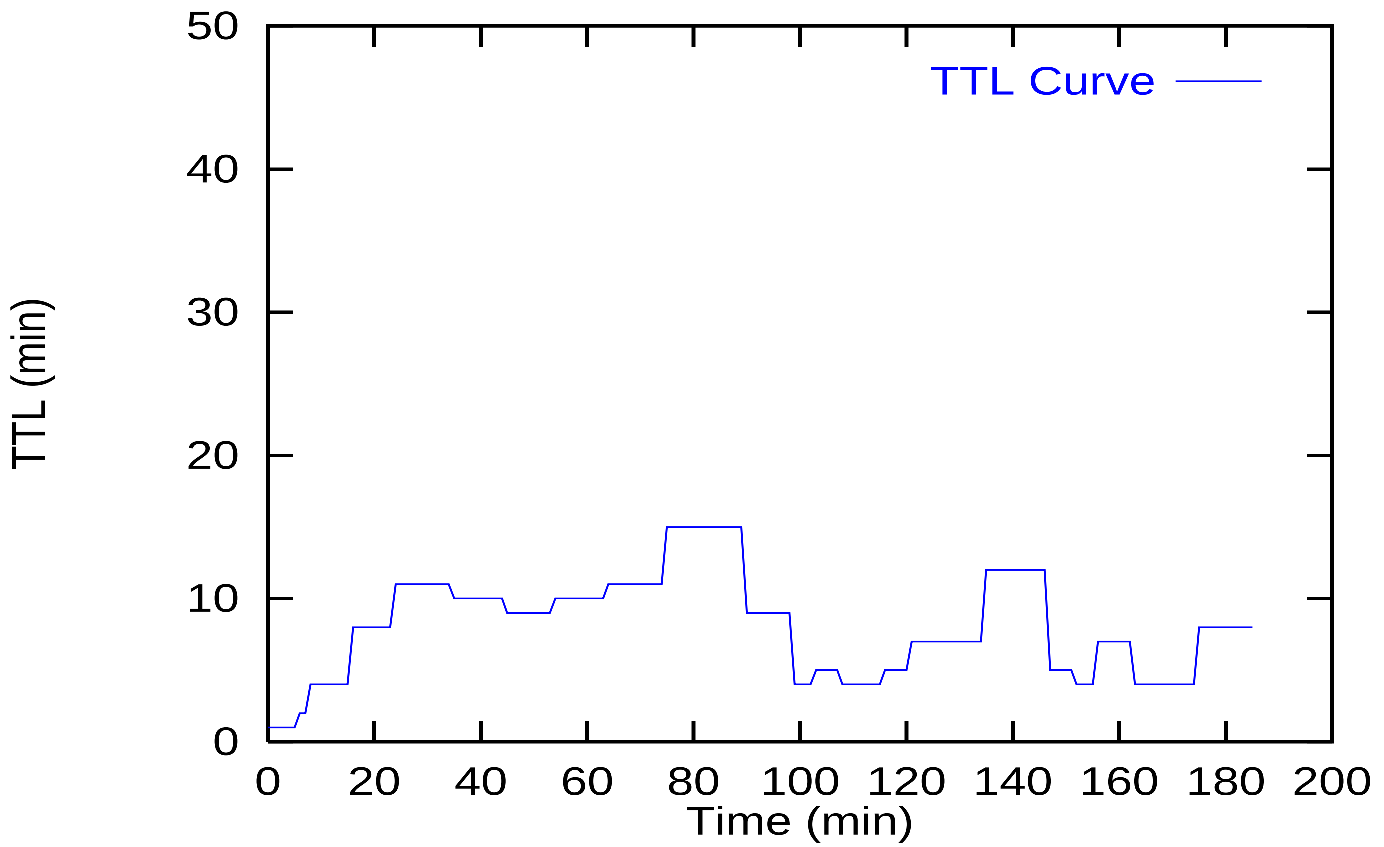
Adaptive TTL – violation probablity



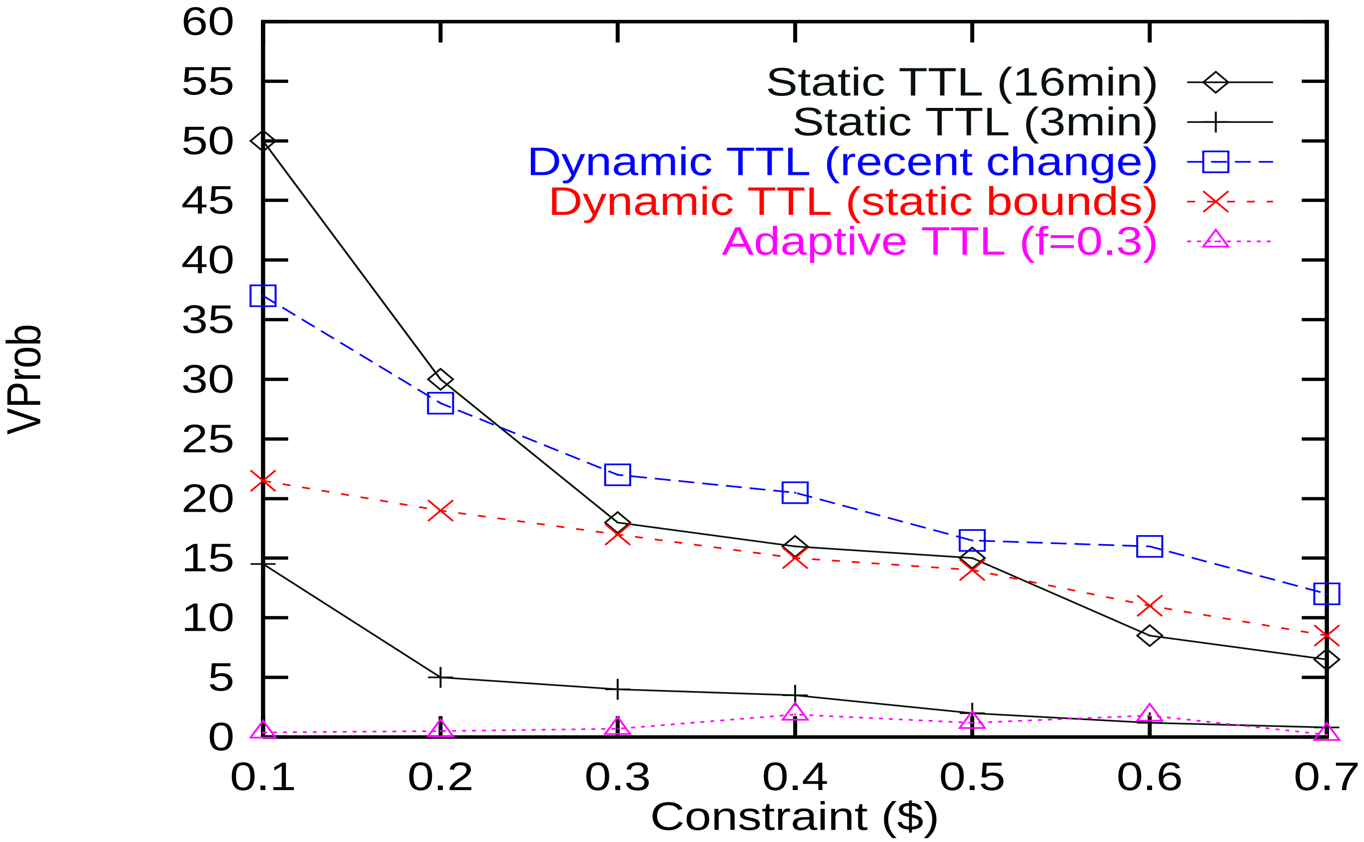
Adaptive TTL – # pollings



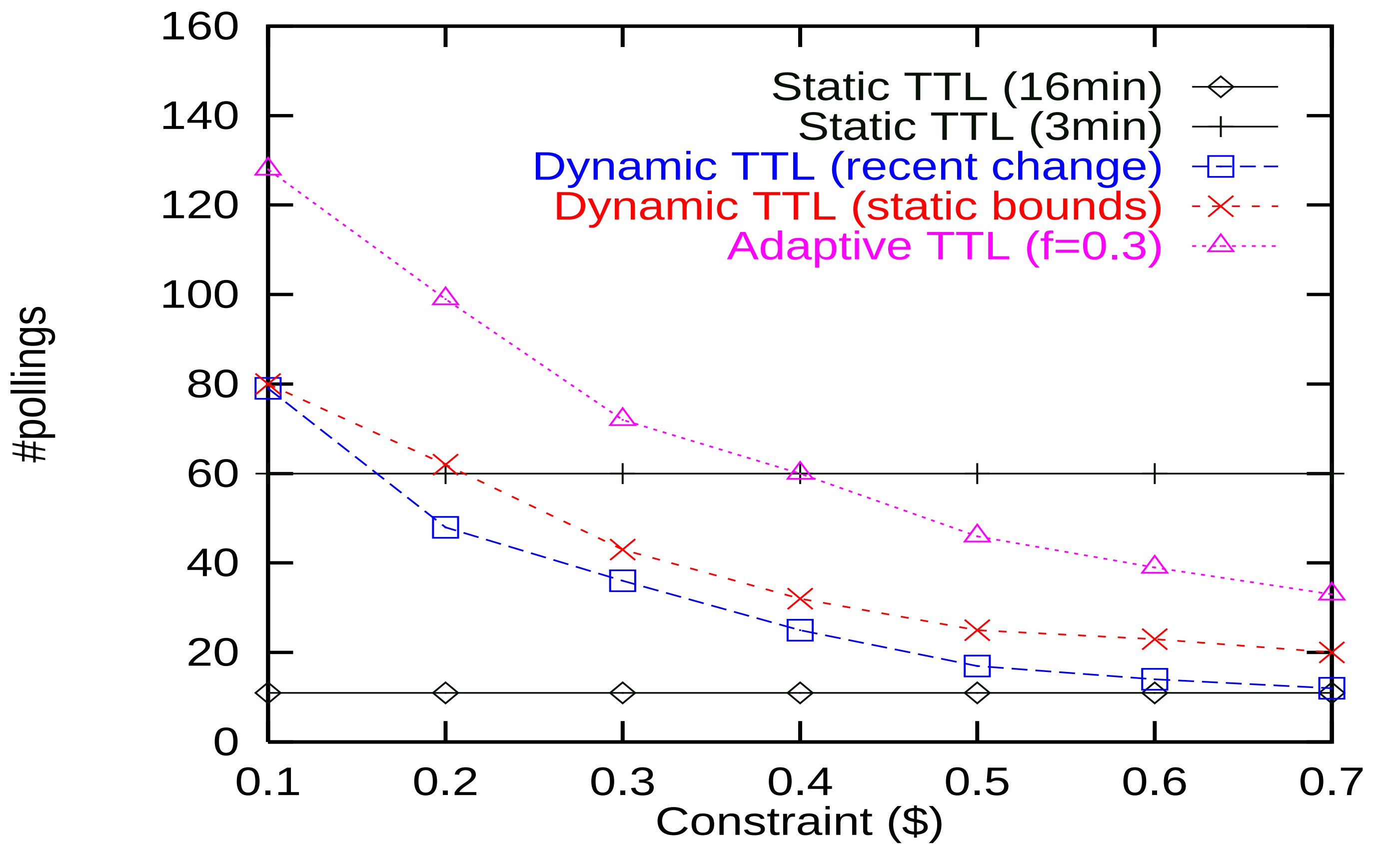
Adaptive TTL – TTL adaptation



Comparison - violation probablity



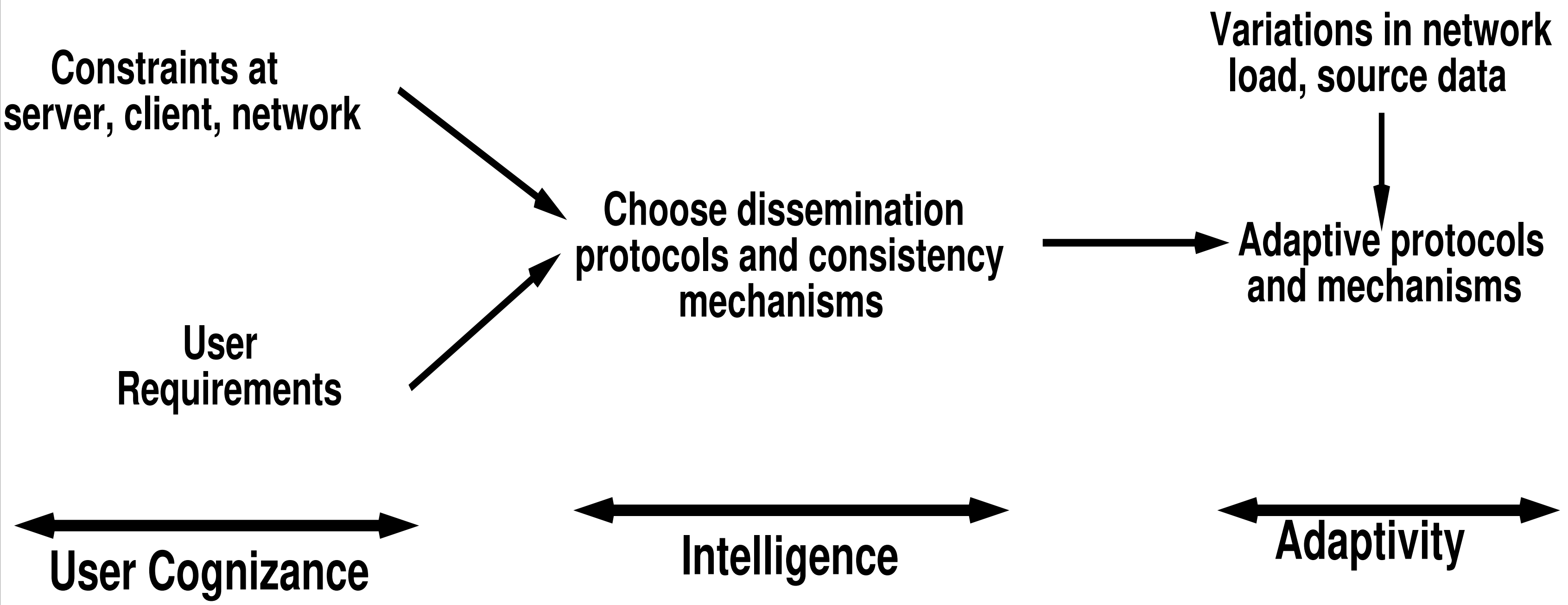
Comparison - # pollings



Summary of Results

- For data whose rate of change is unpredictable, best to use an adaptive algorithm.
- Recent trend *and* past trend must be taken into account.
- Use an upper bound to limit TTL.
- TTL should be tighter when the source changes faster than before.
- Limit the rate at which TTL can increase.

Future Work


Constraints at server, client, network
User Requirements
Choose dissemination protocols and consistency mechanisms
Variations in network load, source data
Adaptive protocols and mechanisms
User Cognizance
Intelligence
Adaptivity