Optimization Technology Survey

Subir Varma

TCP Optimization

TCP Optimization

- Requirements
  - The TCP optimization code should run in the proxy, and should not require any changes in either the server or client TCP stack
  - The design should shield the connection from abrupt changes in the Round Trip Delay (caused due to ARQ, LKAD and/or handoffs)
  - The design shield the connection from abrupt changes in bottleneck node service rate (caused due to ARQ, LKAD and/or handoffs)
  - The design should minimize the amount of buffering (per flow), that is required in the ASN GW
  - The design should enable dynamic re-location of control and data (in the TCP proxy) in response to client mobility
  - Other requirements:
    - The design should enable Freezing of TCP connections in response to a link disconnection
    - The design should allow for a Large Initial Window Option to account for large Delay Bandwidth product
    - The design should support the Time Stamp Option to guard against Spurious Timeouts and Spurious DupAcks
    - The design should support ECN (Explicit Congestion Notification)

TCP Optimization (cont)

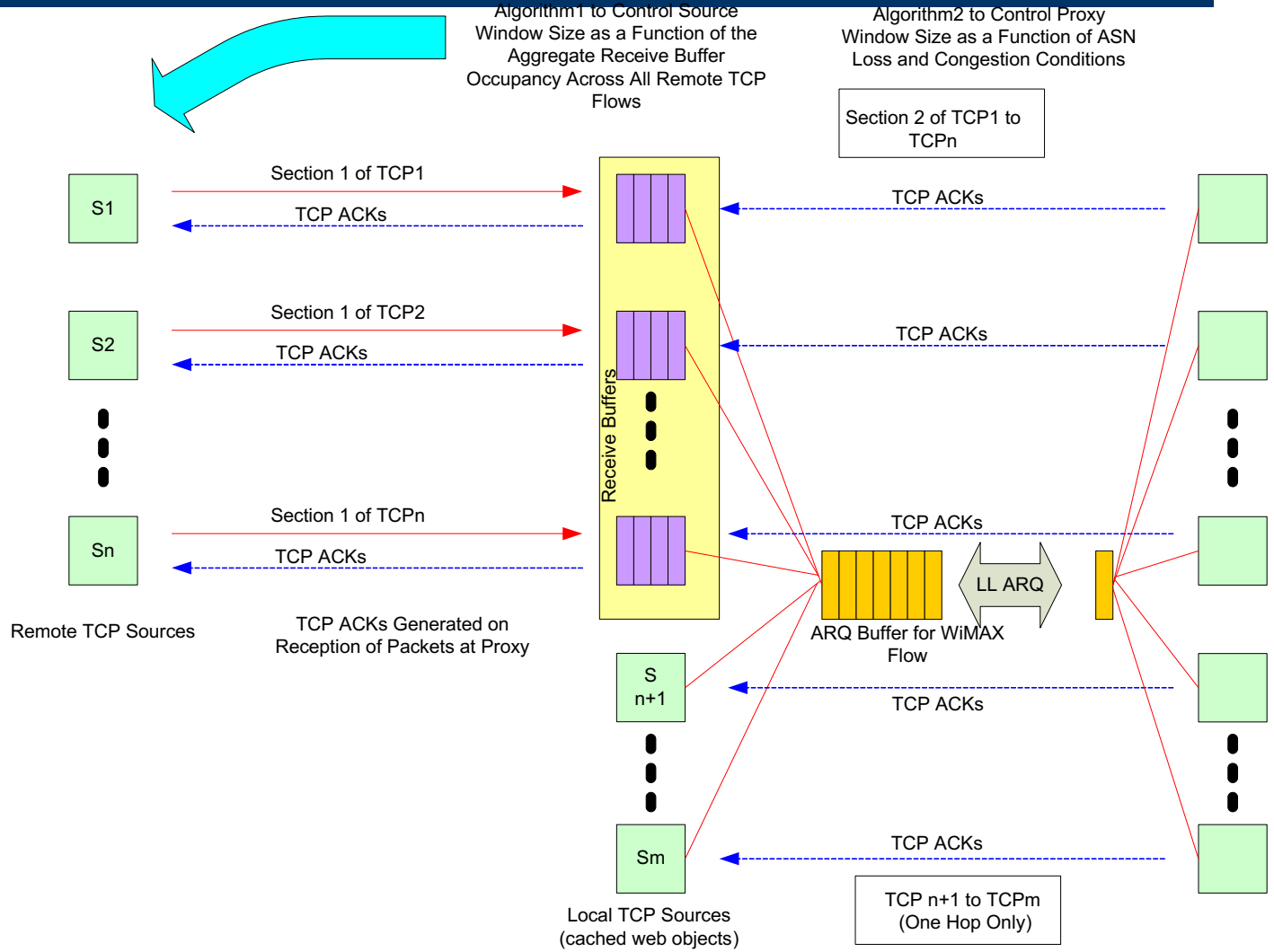
Proxy based TCP optimization should be done at the edge for the following reasons:

- The TCP proxy module at the edge can distinguish between the following types of packet losses:
  - Congestion → Reduce transmit rate
  - Link errors → Retransmit without reducing rate
  - Handoffs → Freeze source and Recover from multiple packet losses without invoking exponential timeouts

  And take appropriate action. In contrast, the TCP algorithm running in the server is not able to do so, and blindly reduces transmission rate and/or invoke exponential timeouts, in response to any type of loss
- We have further enhanced the TCP algorithm by the following:
  - Proactive detection of congestion, without having to wait for packet loss
  - Explicit calculation of the bottleneck link rate, which has several benefits

High Level Design

Two Key Algorithms



Q&A

- **What are the benefits of splitting the TCP connection?**
  - It allows the proxy to use the extra information that it has access to (RF and link congestion conditions) to control and optimize TCP performance, on both sides
  - The pipelining effect created due to the split connection boosts the speed of the end to end connection, by reducing the delay BW product over each link
  - It shields the core network from vagaries of the access network and vice versa
  - It meshes well with the proxy caching architecture. Since TCP connections serving the proxy cache are created locally, all TCP connections over the access network originate at the proxy, whether local are remote
  - It allows the proxy to tailor the behavior of the TCP connection over the ASN to better suit local conditions in the access network
  - It allows the proxy to “freeze” the TCP source in response to temporary link disconnections

Q&A (cont)

- **What are the benefits of having a sophisticated receiver window control at the proxy (aka Algorithm 1)?**
  - It allows the proxy to reduce the amount of packet buffering, which has a number of advantages for the connection (reduced RTT, less data to transfer to another ASN in case of handoff etc)
  - Avoids buffer overflows and underflows at edge router.
  - It makes TCP performance independent of RTT, and connection duration
  - It allows the sending TCP rate to react quickly to variations in service rate and delay (over the wireless link). Reduces number of TCP timeouts triggered by link variation

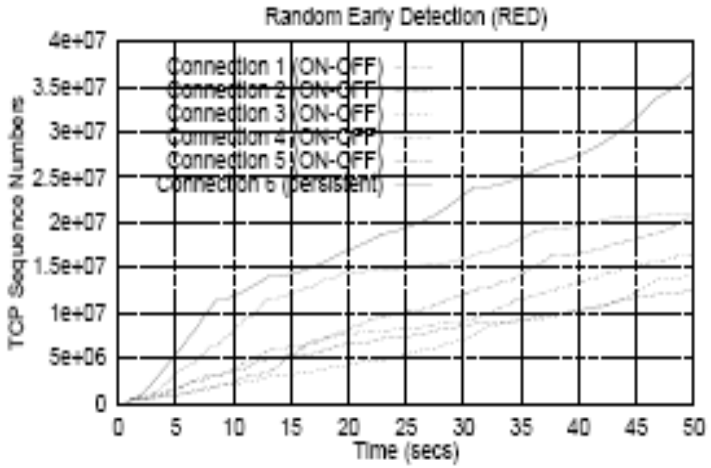
- **What are the benefits of a specially tuned TCP connection over the ASN (aka Algorithm 2)?**
  - It leads to faster recovery from timeout events
  - Allows the algorithm distinguishes losses due to congestion from that due to RF conditions and handoffs, thus reducing the occurrences of congestion control and improving performance
  - It uses rate estimation to optimally control the TCP algorithm
  - It allows all the TCP connections on the access flow to share a common set of TCP parameters (cwnd and RTO), which leads to improved performance

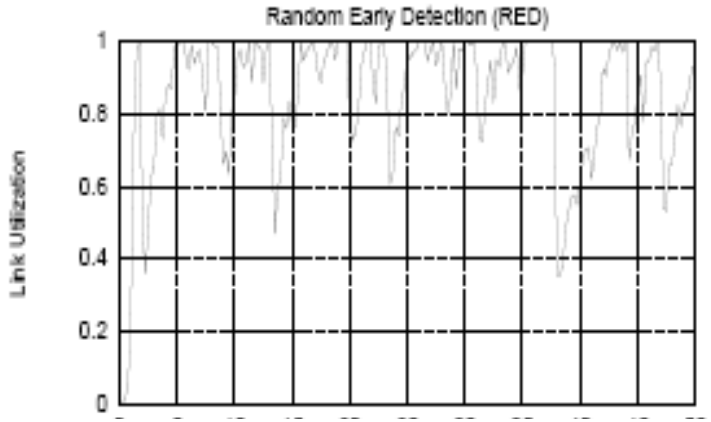
Algorithm 1 Alternatives

1. Random Early Detection (RED)
   - Was designed in the early 90s, has been incorporated into most edge routers
2. Explicit Congestion Notification (ECN)
   - An improvement to RED, requires modifications at the source and receiver, in addition to the router. Has not been implemented very widely in servers.
3. Explicit Window Adaptation (EWA)
   - Was designed to control the buffer in edge routers located between IP and ATM networks. Was shown to perform better than RED.
4. CLAMP
   - An algorithm designed by researchers at the University of Melbourne. Has been implemented as part of the GPRSWeb project
5. Window Regulator with ACK Buffer (WRB)
   - From Ramjee at Lucent. Unlike options (1) to (4), requires per TCP connection processing at edge router

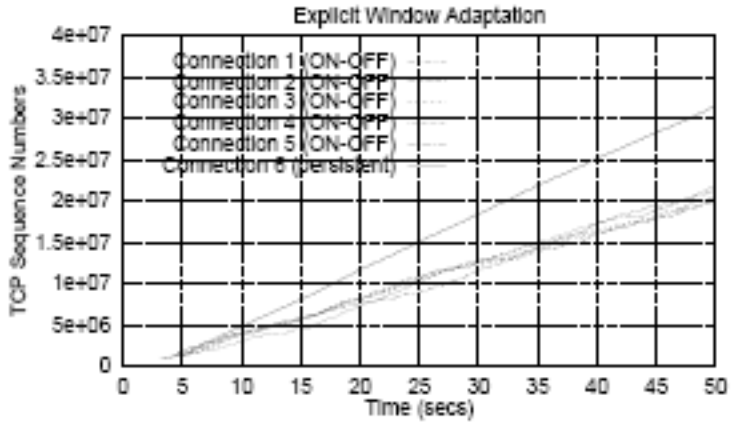
Examples of Performance Improvement (EWA vs RED): 10 Persistent Sources and 30 ON-OFF Sources

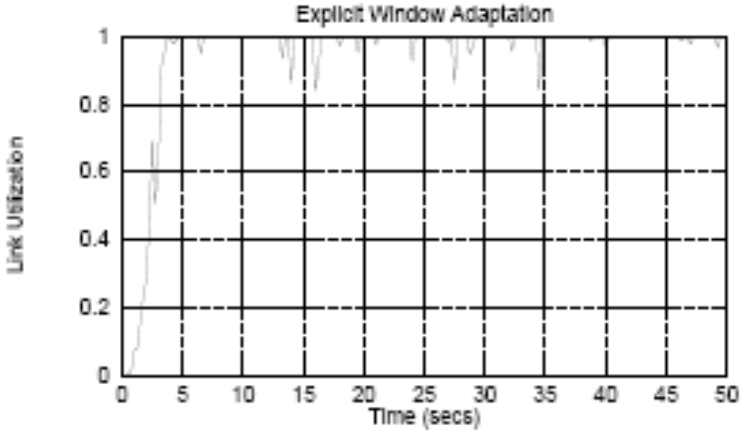
on in
growth,
fairness
nections



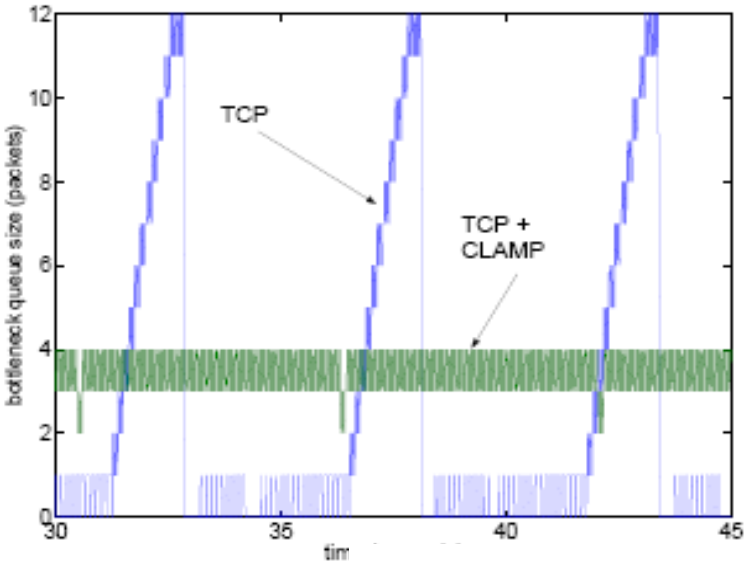


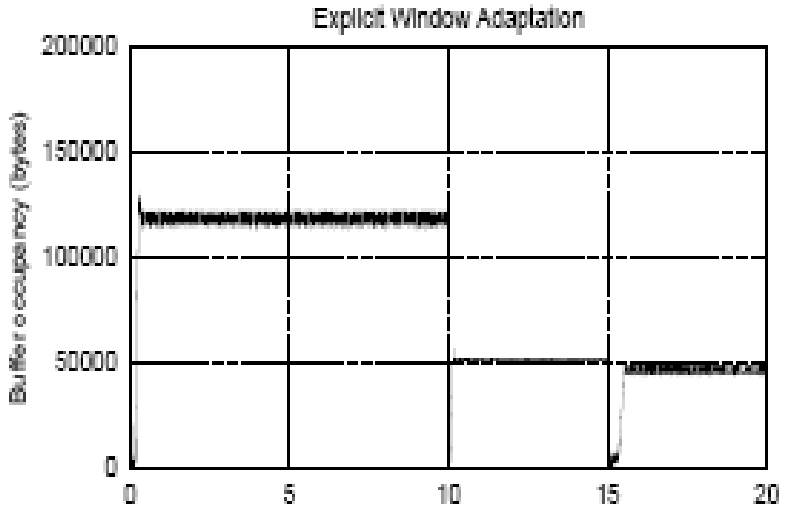
Shows var
link utlizati
TCP timeo

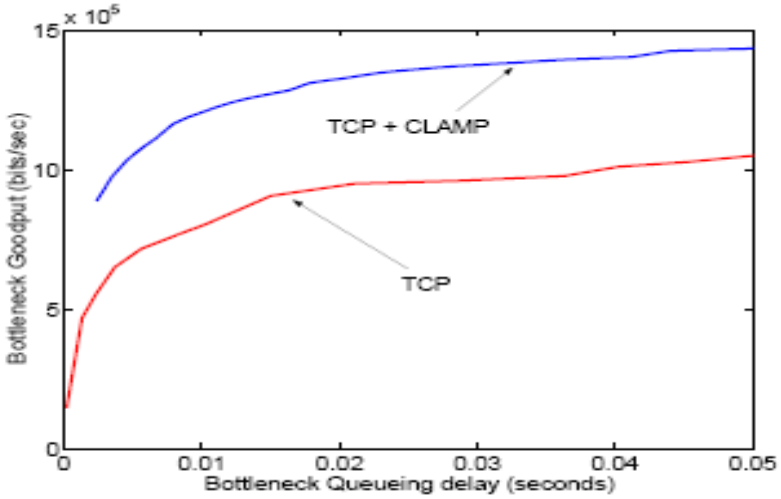




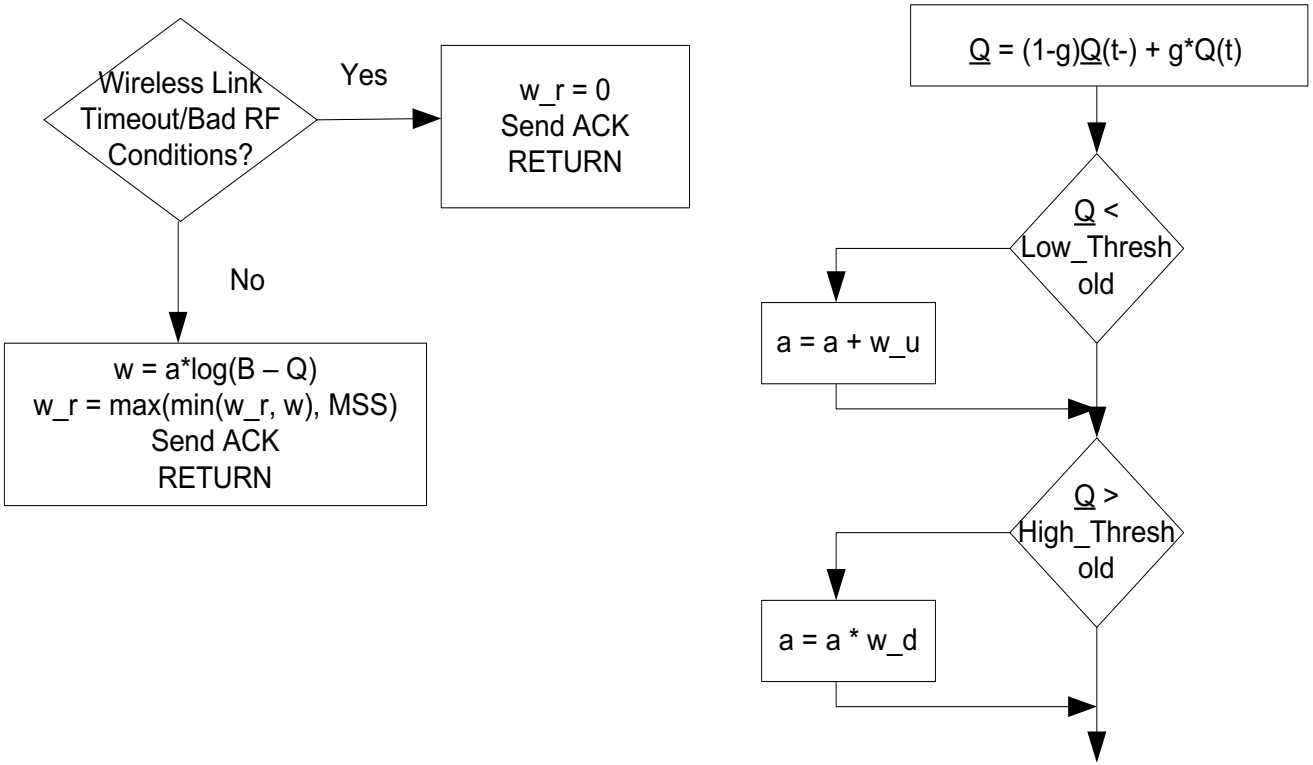
Examples of Performance Improvement (EWA, CLAMP vs Ordinary TCP): Variation in Edge Router Buffer Size



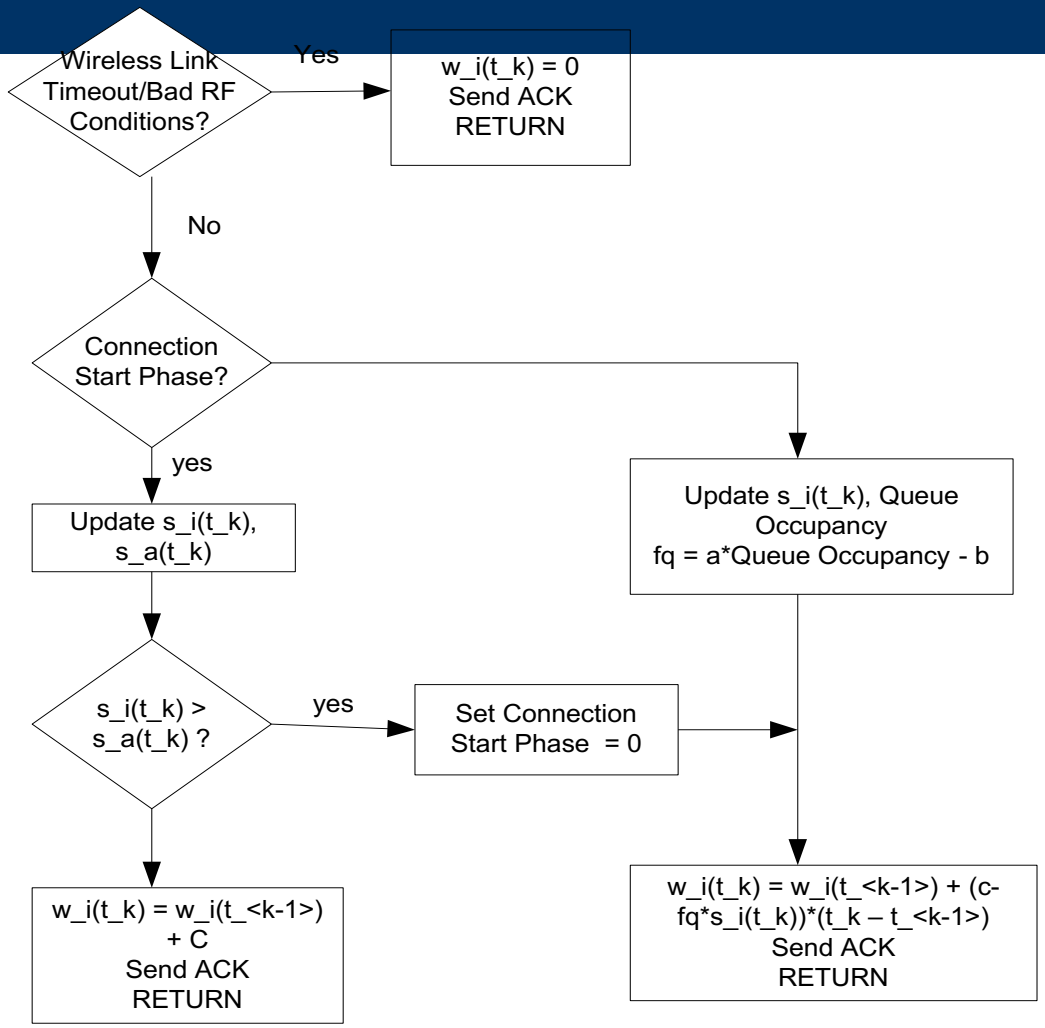




Algorithm 1: EWA Based



Algorithm 1: CLAMP Based

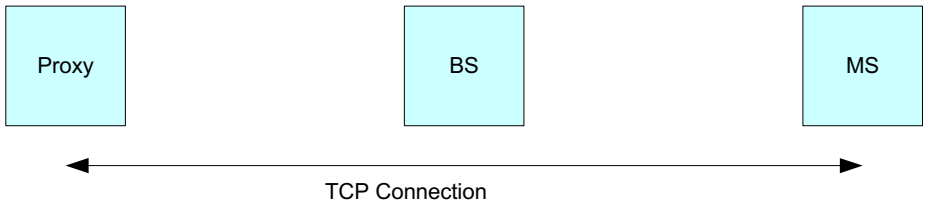


Handling TCP ACKs in Algorithm 1

- TCP ACKs for Hop 1 should be sent back when the corresponding TCP packet arrives at the proxy
- Investigate the feasbility of holding ACKs back, in case there is a decrease in TCP window size, these ACKs are transmitted when a new packet causes an increase in TCP window size
- If packets are dropped at the buffer due to an overflow, the proxy receiver should make sure that this is reflected in the ACK flow (so that the source re-transmits the packets)

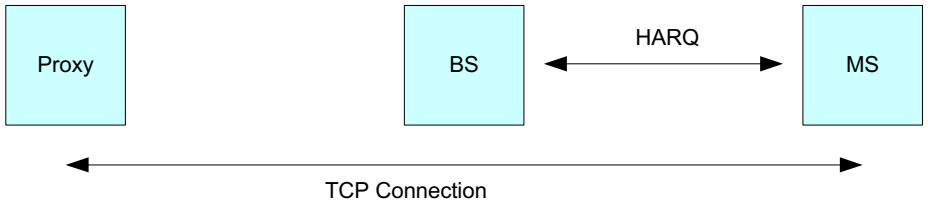
Different Scenarios for TCP Algorithm 2 and WiMAX ARQ Interaction

Case 1



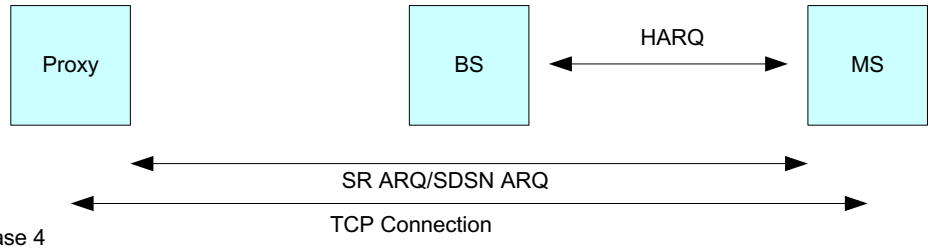
-No Link Level ARQ
- TCP2 used to recover packets lost
Due to congestion + Due to RF impairments
+ Due to Handoffs

Case 2



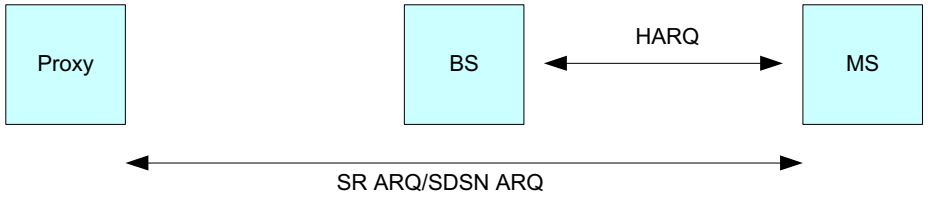
Case 2
-HARQ used to recover packets lost
Due to RF impairments
- TCP2 used to recover packets lost due
To congestion + Due to Handoffs

Case 3



Case 3
- HARQ used to recover packets lost
due to RF impairments
- SR/SDSN ARQ used to recover
packets lost due to Handoffs and
congestion at BS
- TCP2 used to recover packets lost due
congestion at Proxy

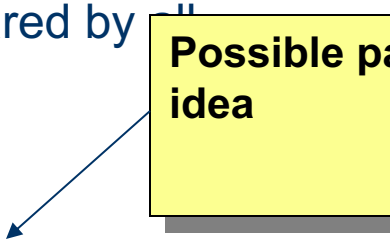
Case 4



Case 4
- No TCP2
- HARQ used to recover packets lost
due to RF impairments
- SR/SDSN ARQ used to recover
packets lost due to Handoffs and
congestion at BS

Algorithm 2

- If a client supports TCP SACK, then this should also be supported by the proxy
- Use a single congestion window and RTO value that is shared by all connections in the flow
- Set the initial cwnd to 4*mss
- Use a large MTU value
- Use a finer grained re-transmission timer
- Distinguish losses due to congestion from that due to RF conditions (using RRM and LL_ARQ information). If the loss is due to RF conditions, then re-transmit packet without invoking congestion control
- After a timeout cwnd is reduced to 1. A single packet is transmitted until ACK received. The timeout value is set to 1.5*RTT, rather than exponential timeout. Once the ACK is received, the cwnd is set back to original size



Algorithm 2 (cont)

- For the TCP link between the proxy and the MS, the fundamental requirement is to be able to distinguish airlink related losses from those due to congestion and handoffs. This is important, since TCP should only change its rate in response to congestion losses.
- Plain TCP (aka Reno) is not able to distinguish between these losses, and hence does not work well over wireless links
- One way to retain TCP Reno over wireless links is to have ARQ over the link layer, so that ARQ is responsible for recovering from link and handoff losses, this is the solution used by 3G and WiMAX. This scheme has the following issues:
  1. ARQ translates wireless losses into variation in link delay. This leads to a phenomenon called ACK Compression, which can lead to bunched packet transmissions resulting in packet loss at the bottleneck
  2. Using ARQ alone in times of link disconnection due to handoffs, can lead to the situation that the TCP source keeps pumping at the high rate, leading to buffer overflow
- The TCP algorithm proposed here gives the operator the option of handling wireless link losses at the TCP layer, without having to rely on LL ARQ. Reducing the number of ARQ layers operating between the proxy and the MS has the following benefits:
  - Helps to reduce variability in TCP RTO estimates, reducing ACK compression and TCP timeouts
  - Shuts down the TCP source during handoffs, thus preventing overload of the ASG or BS buffers
  - Helps to reduce complexity of BS, thus enabling lower cost BS architectures

Elements of Algorithm 2

- Available BW Estimation (ABE)
  - The proxy keeps an estimate of the available BW (between the proxy and the MS), for each TCP flow. The ABE for a flow may fluctuate for the following reasons:
    - Variation in the BW allocated to a SF by the proxy and/or the BS. This is caused due to the presence of other flows which are competing for the same BW
    - Variation the BW allocated to a MS by the BS, due to the LKAD algorithm adjusting the BW as per the current link SNR
    - Re-transmissions at the link layer cause the available BW to decrease
- The ABE is also useful during Dynamic Content Adaptation to tailor the streaming media rate to the available link bandwidth

ABE Algorithm

```
If (Packet is sent)
   sample_length(k) = packet_size*8;
   sample_interval(k) = now - last_sending_time;
   Average_packet_length(k) =
       alpha*Average_packet_length(k-1) +
       (1 - alpha)*sample_length(k);
   Average_interval(k) = alpha*Average_interval(k-1) +
       (1 - alpha)*sample_interval(k);
   ABE(k) = (1-exp(-T(k)/T_0))*
          Average_packet_length(k)/Average_interval(k)
         + exp(-T(k)/T_0)* ABE(k-1);
Endif
```

Uses for the ABE Value

- Detection of Link Congestion:
  Compute the following
  `Virtual Queue Length = cwin – ABE*RTT_min`
  If the VQL exceeds a threshold, then the proxy should initiate congestion control
- Rate Control Algorithm

```
ssthresh = ABE * RTT_min
if (cwnd > ssthresh)
        cwnd = ssthresh
endif
```

Procedure on Receiving ACK

```
ABE( )                    /* Invoke bottleneck BW computation procedure */
VQL = cwin - ABE*RTT_min  /* Compute bottleneck queue length estimate */

If (ACK and VQL < Threshold)  /* Normal TCP Reno behavior */
    SS() or CA()
Endif

If (ACK and VQL > Threshold) /* Proxy detects congestion building and
    Rate_Control()             takes pre-emptive action by
    SS() or CA()                reducing rate */
Endif

If (DUPAck and VQL < Threshold) /* Since VQL is under the threshold
    explicit_retransmit()             the packet loss is due to link error or handoff
    fast_recovery()                    hence proxy re-transmits packet w/o
Endif                                  invoking congestion control */

If (DUPAck and VQL > Threshold) /* Since VQL is over threshold
    Rate_Control()            the packet loss is likely due to
    explicit_retransmit()     congestion, hence proxy invokes
    fast_recovery()           congestion control */
Endif
```

Link Disconnection Events

Causes of Link Disconnections

- Fully Controlled Handoff
  - Detected by presence of HO_REQ message from Serving BS (Profile C) and a subsequent DP PRE-REG REQ from the Target BS
- Uncontrolled Handoff
  - ACKs stop coming from the MS
- Link Fade
  - ACKs stop coming from the MS

Can Link Disconnections due to Handoffs be made transparent to the TCP layer, by using link layer Data Integrity techniques (buffering or bi-casting)?

Reaction to Link Disconnection Events (In Absence of LL Data Integrity)

- TCP2 reaction to Link Disconnection event:
  - Freeze TCP2 connection by reducing its cwnd to zero
  - Communicate the event to TCP1

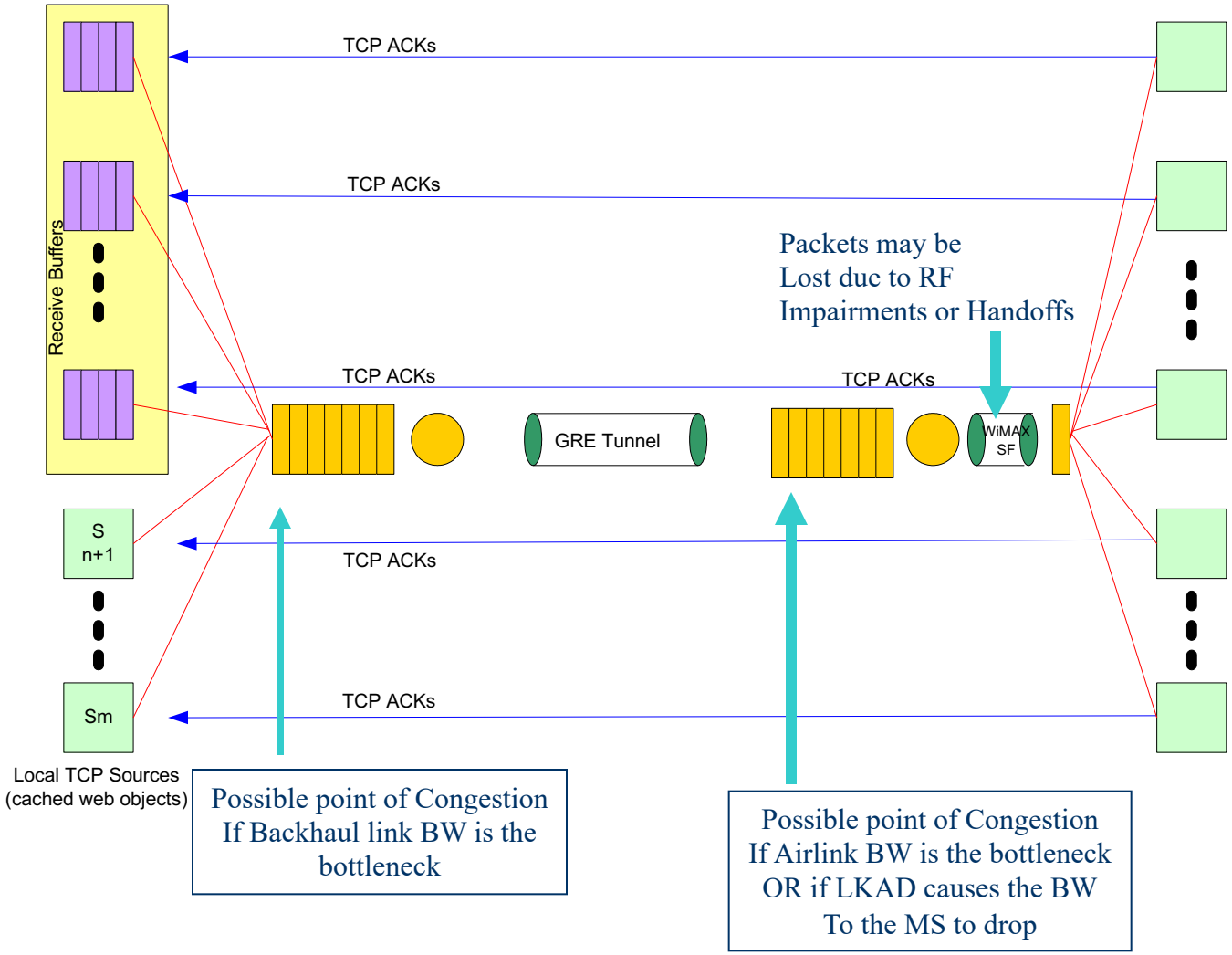
  Note: If multiple (un ACK'd) packets are in transit at the time the disconnection occurs, then this cause exponential timeout(s) with TCP Reno type algorithms (which can recover only a limited number of packets without timeouts). TCP2 should be capable of re-transmitting multiple packets lost due to handoffs, without invoking congestion control.
- TCP1 reaction to link Disconnection event
  - Set cwnd to zero in the next ACK it returns
- TCP2 reaction to link re-connection event
  - Set cwnd back to value before disconnection and re-start transmissions
  - Communicate event to TCP 1
- TCP1 reaction to link re-connection event
  - Send DUPACKs back to server to restart traffic flow

Case 1: TCP2 without any Link Level ARQ

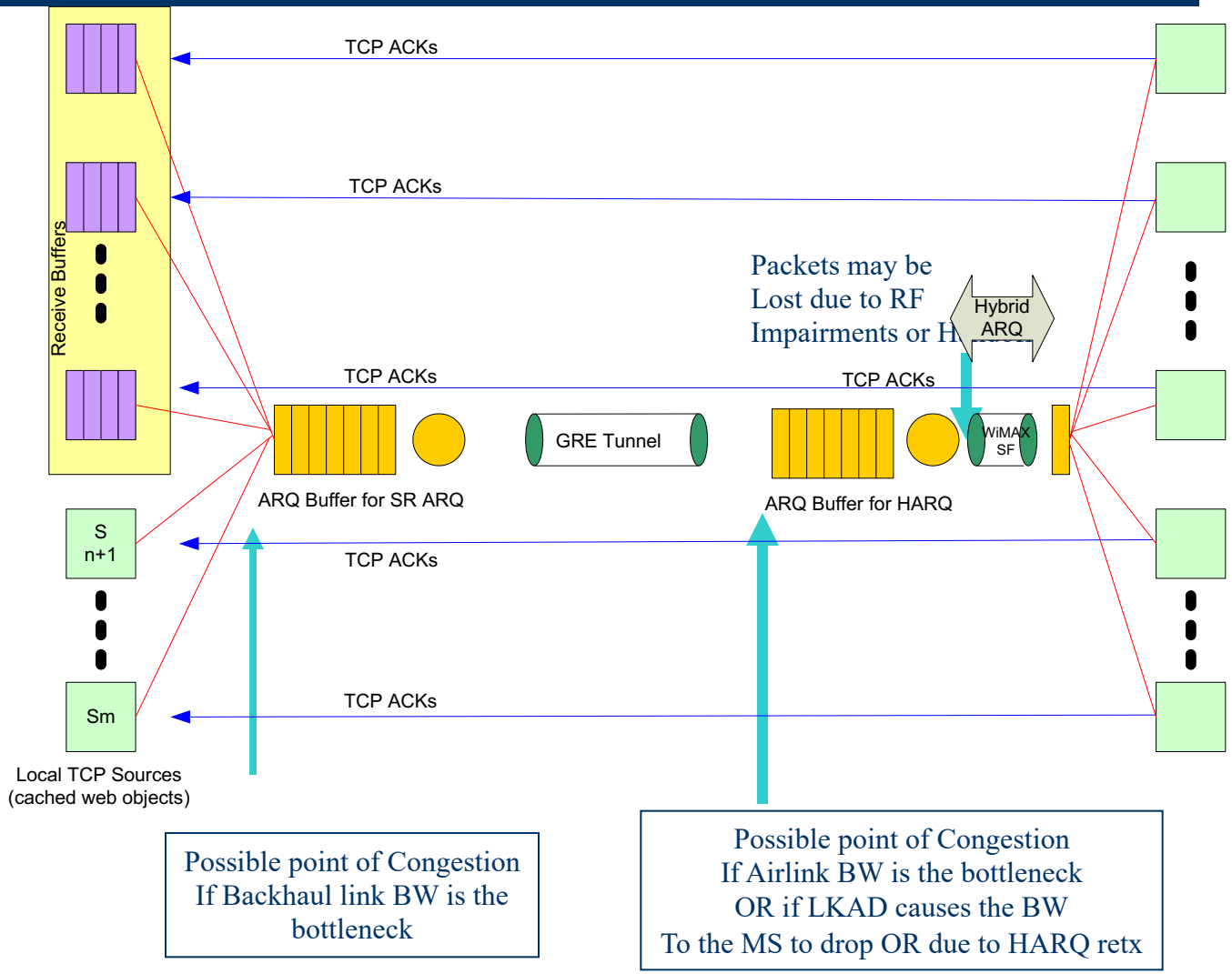
Requirements on TCP2 to be able to function effectively without any Link Level ARQ:

- Be able to distinguish losses due to Link Congestion from losses due to RF Impairments and losses due to Handoffs
- If the loss is due to RF Impairments, then TCP2 should re-transmit the packet without invoking congestion control
- If the loss is due to Link Congestion at the BS then TCP2 should invoke congestion control (and also re-transmit the missing packet)

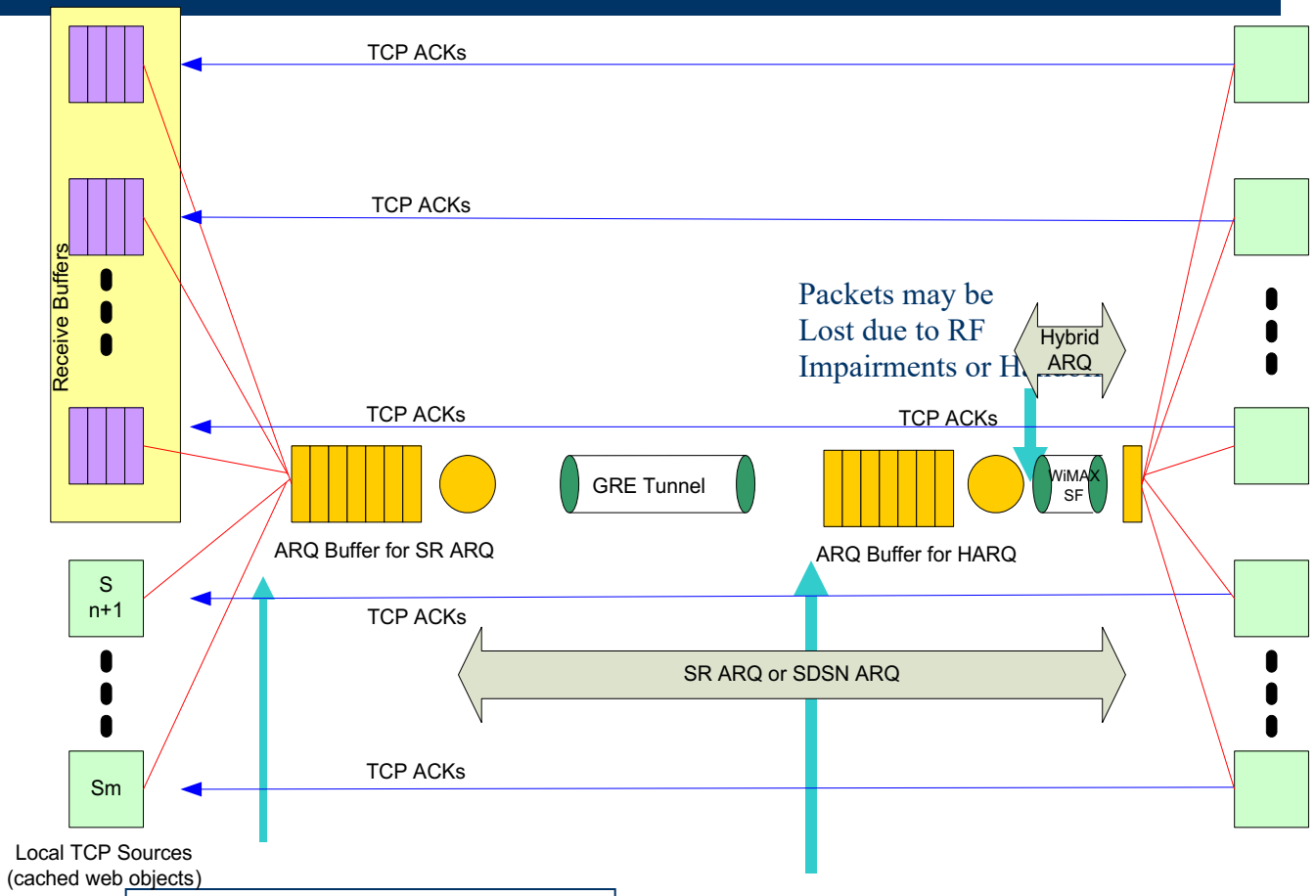
Case 1 (cont)


TCP ACKs
TCP ACKs
Receive Buffers
Packets may be
Lost due to RF
Impairments or Handoffs
TCP ACKs
TCP ACKs
GRE Tunnel
WiMAX
SF
S
n+1
TCP ACKs
Sm
TCP ACKs
Local TCP Sources
(cached web objects)
Possible point of Congestion
If Backhaul link BW is the
bottleneck
Possible point of Congestion
If Airlink BW is the bottleneck
OR if LKAD causes the BW
To the MS to drop

Case 2


TCP ACKs
TCP ACKs
Receive Buffers
Packets may be
Lost due to RF
Impairments or H
Hybrid
ARQ
TCP ACKs
TCP ACKs
GRE Tunnel
WiMAX
SF
ARQ Buffer for SR ARQ
ARQ Buffer for HARQ
S
n+1
TCP ACKs
Sm
TCP ACKs
Local TCP Sources
(cached web objects)
Possible point of Congestion
If Backhaul link BW is the
bottleneck
Possible point of Congestion
If Airlink BW is the bottleneck
OR if LKAD causes the BW
To the MS to drop OR due to HARQ retx

Case 3



Sole point of Congestion
If Backhaul link BW is the
Bottleneck OR due to SR ARQ retx
OR due to HARQ retx

Interactions with Mobility

- For TCP connections originating at the proxy (and serving the procy cache), it is not possible to migrate these to the target ASN GW while the connections are still open
- This implies that all such connections have to close before the anchor can move from the S_ASN_GW to the T_ASN_GW. This is unlike the behavior for remote TCP connections, that can be migrated while they are still open
- Possible scenario: The MS makes additional HTTP requests after moving to T_ASN_GW, but while still anchored by the S_ASN_GW. How should these be handled?
  The solution involves interaction to the proxy cache and will be discussed in the next section

Web (HTTP) Caching

Web and Streaming Media Caching

- Problem: The user should perceive a LAN like performance over his or her mobile device. Thus the system should hide the “defects” introduced due to operation over a mobile WAN infrastructure, such as High Latencies and BW shortages
- State of the Industry: Proxy based Caching is a standard solution that is used to improve performance for WAN applications and make them more LAN like. However caching has only been done in the context of the caching proxy node remaining fixed (even though the client is moving)
- Solution: Introduce proxy caching in the ASN gateway, which is a natural edge aggregation point for this function. More ever, since the ASN GW may change due to client mobility, we have devised algorithms to manage the cache contents to take this into account.

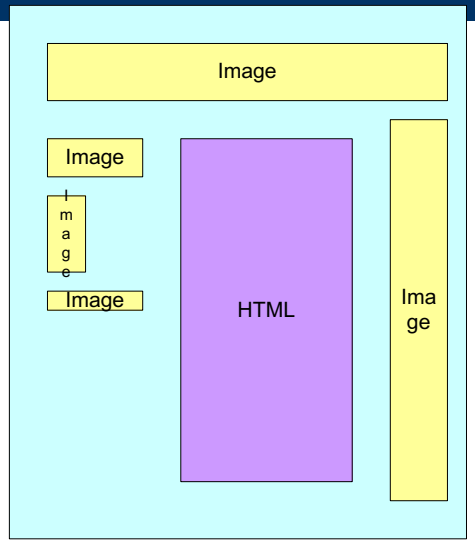
Web Caching

- Requirements:
  - The design should support Content Hash Key (CHK) based caching (enables multiple URLs pointing to identical documents to be stored just once in the cache)
  - The design should support Delta Encoding for Dynamic Web Pages (acts as a compression technique to reduce traffic between the proxy and the client)
  - The design should support the Parse and Push mechanism for Dynamic Web Pages
  - The design should support Content Pre-Fetching (in order to improve the hit rate)
  - The design should support Dynamic Re Location of Cache content (on a per user basis), in response to client mobility
  - The design should support Dynamic Load Balancing across multiple caches
  - The design should support the Edge Side Caching (ESI) protocol (www.esi.org) for caching of Dynamic Web pages
  - The design should take full advantage of any caching that is available at the client device, to further reduce network traffic
  - The design should facilitate co-operation between caches located in multiple ASN GWs. The feature may be especially useful due to mobility (Possible Patent → Send enquiries to only those proxies where the MS was located in recently)
  - The design should support the URL Rewriting/DNS Rewriting techniques

Web Page Composition

tistics taken from
busiest 3000 sites
a large server farm
sting over 34K
mmercial web sites

03)



Object type	Prevalence
Images	77%
HTML	5.2%
CGI	8.6%
.doc,.pdf,.ppt,.ps	0.2%
Audio	0.1%
Other	9.1%

Table 2: Request object type popularities.

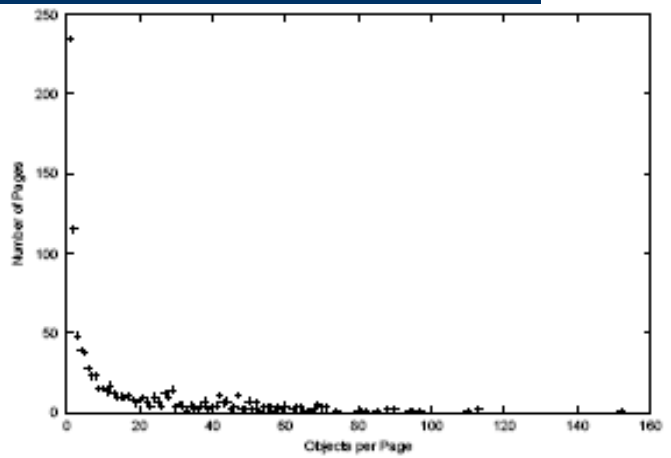
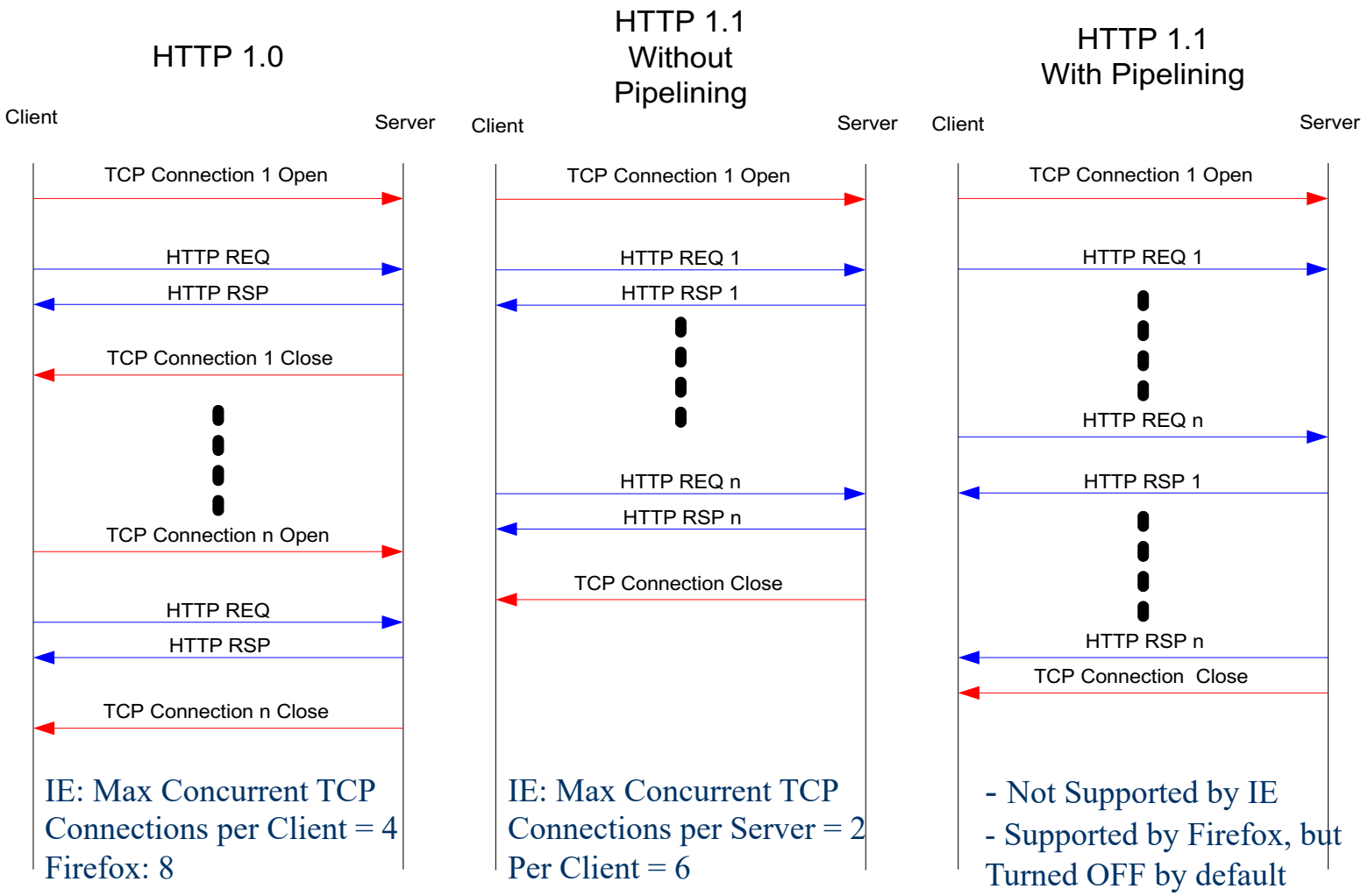


Figure 1: Distribution of objects per page.

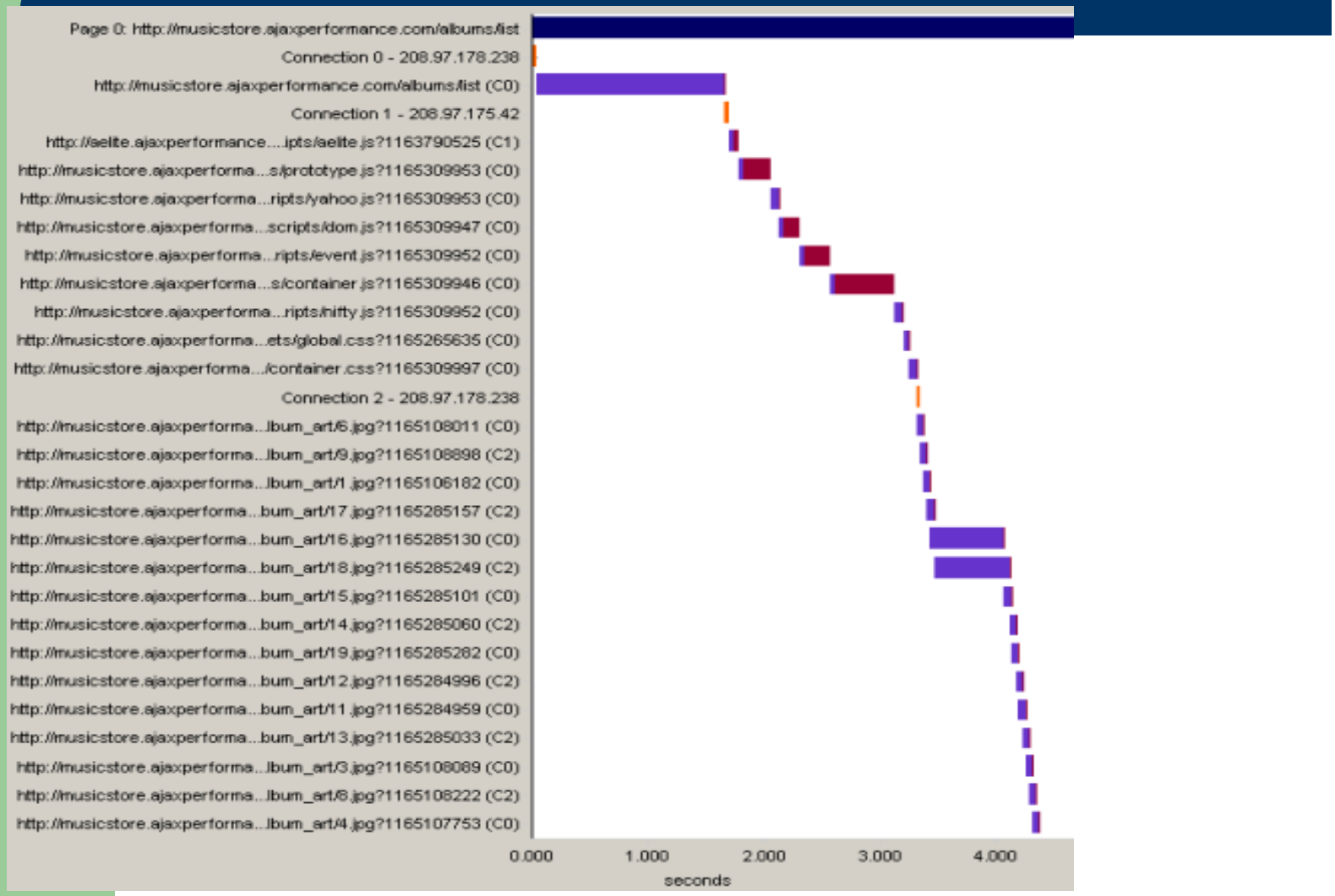
Objects per Page	No. of Pages	No. of Objects
1	235 (25.5%)	235 (1.71%)
2-5	240 (26.1%)	720 (5.24%)
6-15	165 (17.9%)	1577 (11.5%)
16+	280 (30.4%)	11215 (81.6%)
Total	920	13747

Table 4: Number of pages successfully downloaded in each page size category. Mean objects per page is 15, median is 5.

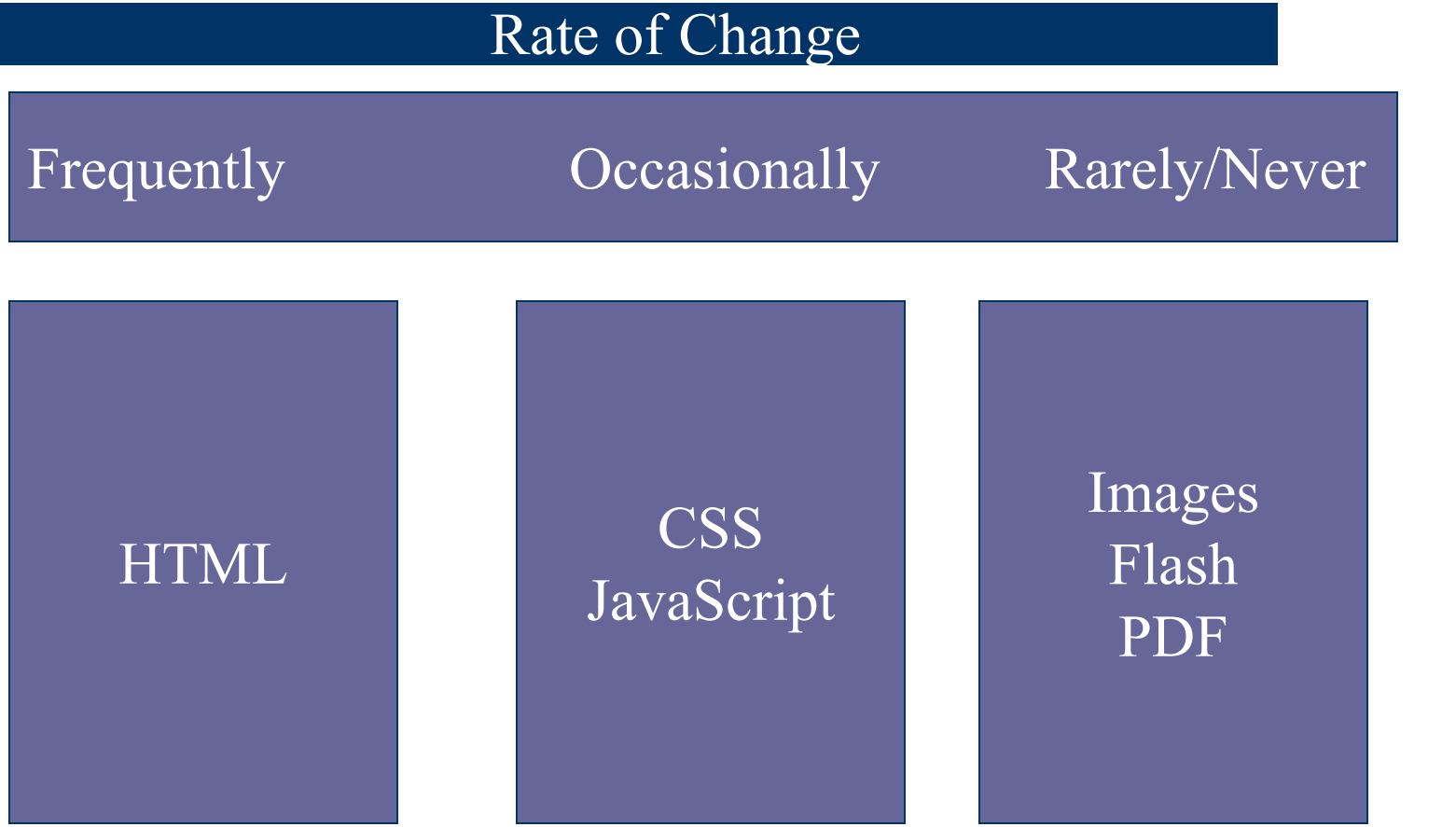
HTTP Operation



Download of a Web Page Using 2 TCP Connections



Common Assumptions about Content Types



Cache Hit Rate as a Function of the Population

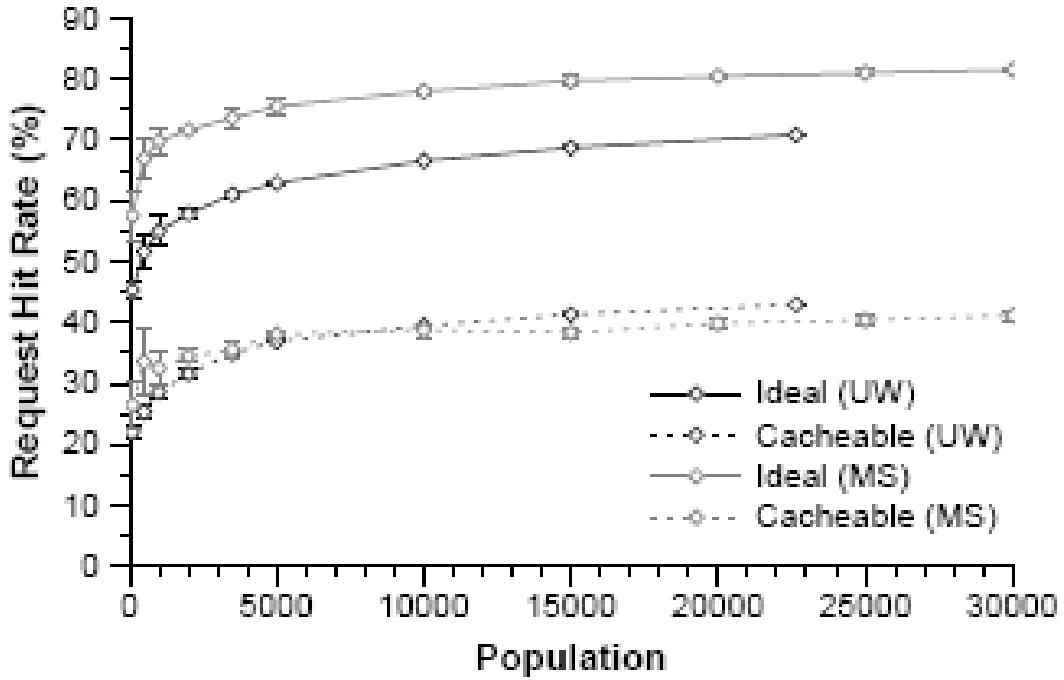
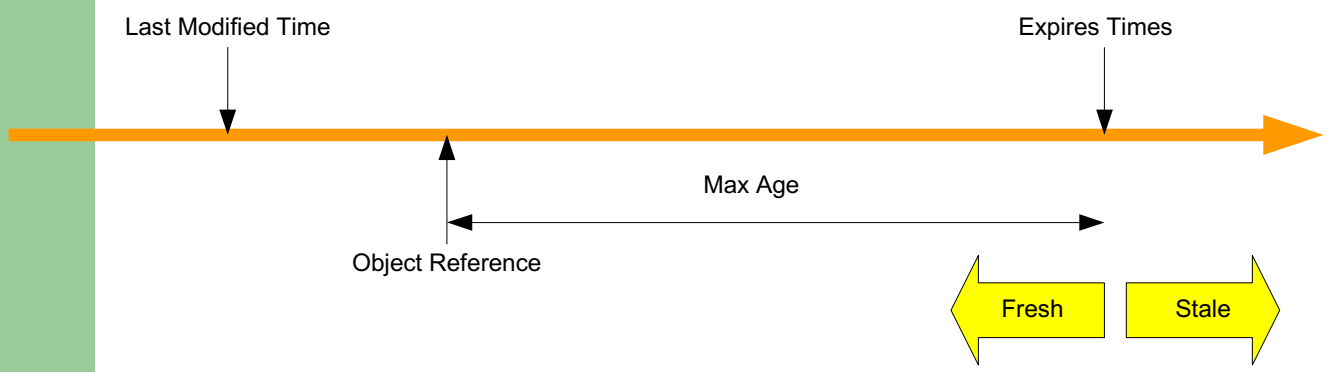


Figure 1. Proxy cache request hit rate as a function of client population.

Occurrence
Of a Knee
At about 2500

Fresh and Stale Cache Entries



Expires or Max Age information is not available in many cases

Example HTTP Response with Cache Control information

```
HTTP/1.1 200 OK
Date: Fri, 30 Oct 1998 13:19:41 GMT
Server: Apache/1.3.3 (Unix)
Cache-Control: max-age=3600, must-revalidate
Expires: Fri, 30 Oct 1998 14:19:41 GMT
Last-Modified: Mon, 29 Jun 1998 02:28:1 GMT
ETag: "3e86-410-3596fbbc"
Content-Length: 1040
Content-Type: text/html
```

Fresh: Content can be served directly from cache

Stale : Cache should validate content with server before serving it

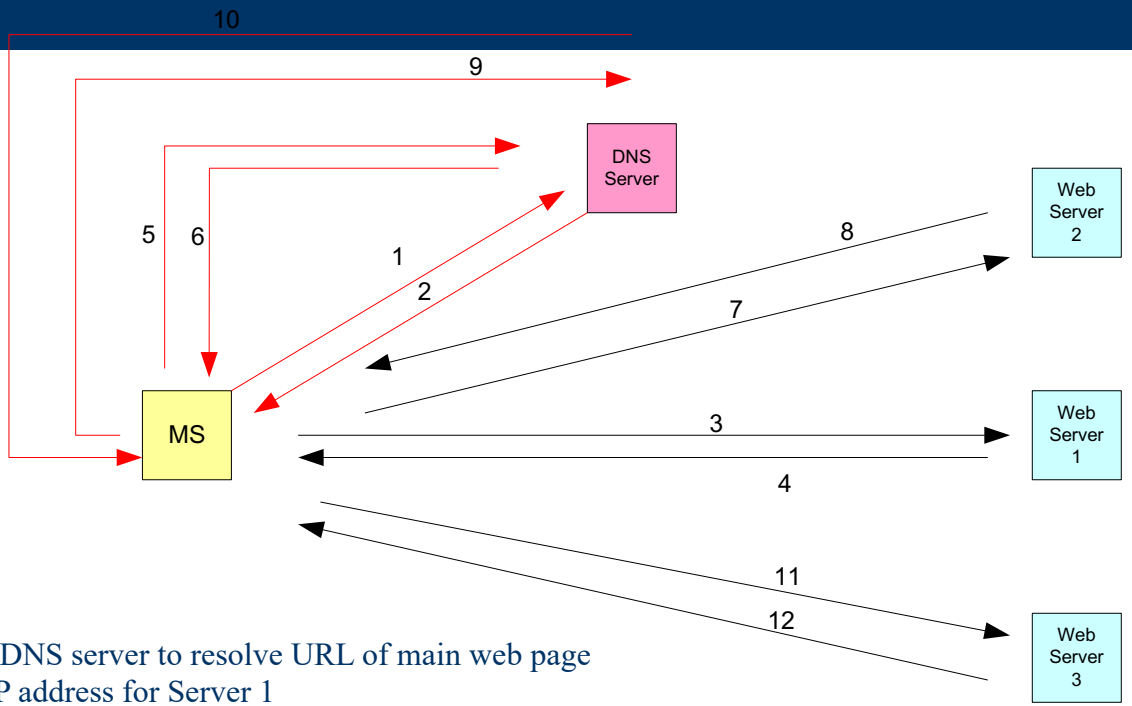
Enhancements on Top of the Basic Squid Caching Algorithm

1. Increase TCP connection parallelism
2. Inter Cache communications
   - Squid supports Internet Cache Protocol (ICP) and Summary Cache Protocol
3. Reduce number of DNS lookups
4. Support for URL Redirect
5. Automatic Freshness Checks and Pre-fetching by Proxy

Requires Client Side Support:

5. Delta Encoding
6. Content Hash based Caching: To mitigate the effects of URL aliasing

Web Page Download: Without Proxy

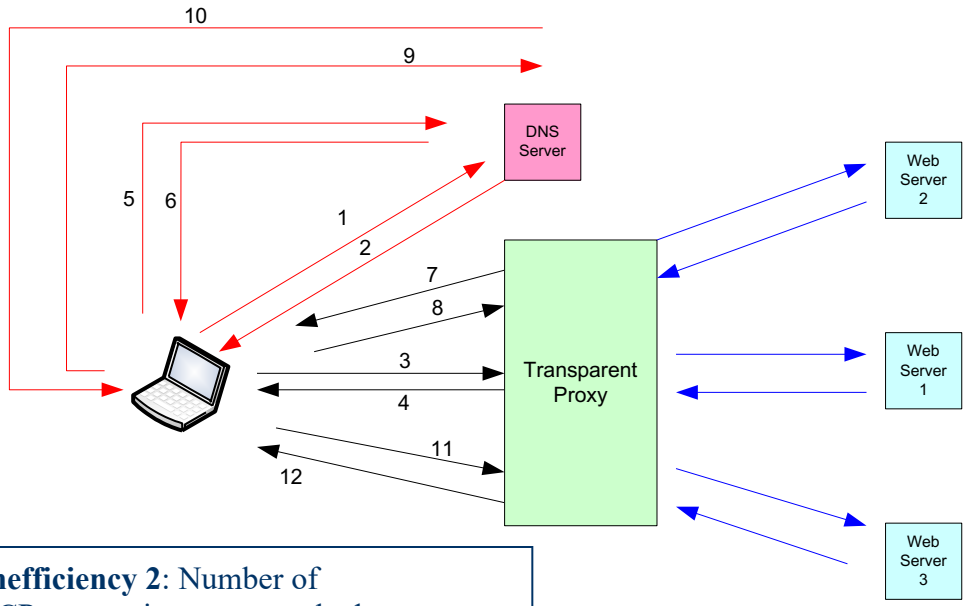


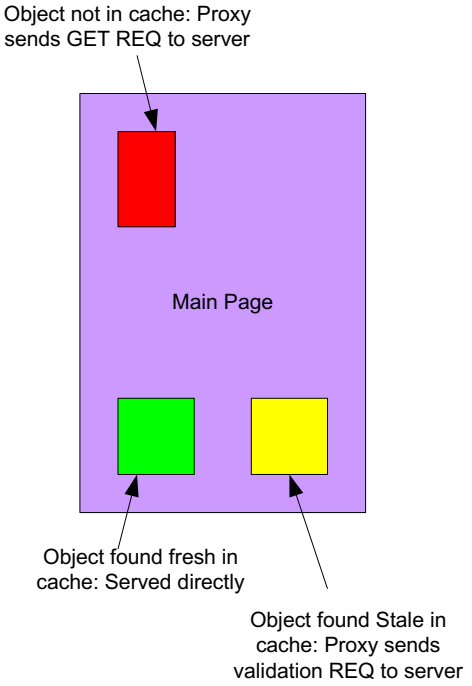
1. MS contacts DNS server to resolve URL of main web page
2. DNS sends IP address for Server 1
3. MS opens TCP connection with Server 1 and sends HTTP GET
4. Server 1 send main web page to MS in HTTP Reply
5. MS parses main page to set URLs of embedded objects. It sends DNS request to resolve address of objects in FQDN2
6. DNS sends IP address for Server 2
7. MS opens TCP connection with Server 2 and sends one or more (sequential) HTTP GETs
8. Server 2 sends embedded objects to MS
9. MS repeats these steps to retrieve objects from Server 3

Web Page Download: With Proxy

Inefficiency 1: Multiple DNS lookups
Required if objects spread over more than 1 FQDNs

Inefficiency 3: Frequent TCP connection
Setup and teardowns between proxy and remote servers





Inefficiency 2: Number of
TCP connections not matched to
The number of objects being downloaded

Inefficiency 3: Client forced to verify
Freshness of objects

Inefficiency 4: As much as 10% of
HTTP server responses are Re-Directs
Forcing the client to make a second request

Benefits of Proxy Architecture:
- Proxy can use pipelined mode to downloads objects from servers
- Proxy can pre-validate and/or pre-fetch stale objects
- Reduced number of connections with client

URL Rewrite (Addresses Inefficiencies 1 and 2)

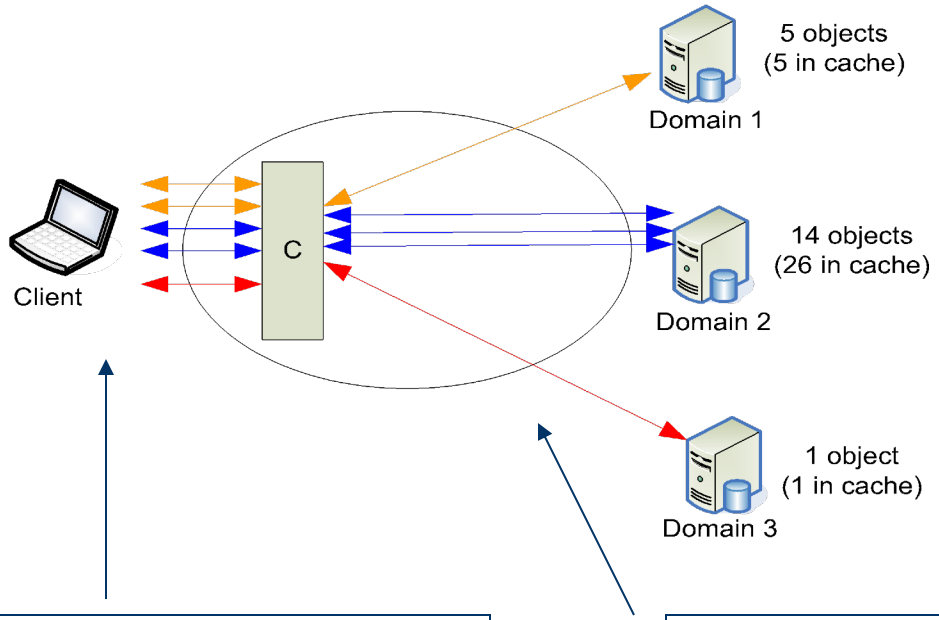
1. Client makes REQ for top level page. The URL for the web server is resolved by the DNS server after which the HTTP REQ is sent by the client and intercepted by the Proxy.
2. The proxy either serves the page from its cache, or in case of a MISS, fetches it from the web server
3. The Proxy parses the HTML, and extracts all the references to embedded objects within it. It then appends a “fixed” IP address to all the URLs, for example http://www.yahoo/com/images/boat.jpg becomes http://10.0.0.9/ www.yahoo/com/images/boat.jpg .
4. When the client is required to fetch this object, it opens a connection to 10.0.0.9 (at the proxy) and requests the URL www.yahoo/com/images/boat.jpg . Note that all embedded objects are now downloaded using up to 2 TCP connections between the client and the proxy
5. The Proxy may use multiple such Pseudo IP addresses in order to increase the number of TCP connections between itself and the client

Note that only 1 DNS lookup was needed to download all the objects in the web page

Addressing the Other Inefficiencies

- Inefficiency 3: Object Freshness Information
  - When the proxy re-writes the URLs in Step 3 (of previous page), it also appends the latest page freshness information. It has previously obtained this information either when updating the object for other clients, or by pro-actively checking the freshness of objects in the cache.
  - If the client has the object in the browser cache, it can look at the freshness information and based on that it may not need to send a HTTP REQ to verify freshness
- Inefficiency 4: HTTP Redirects
  - A sizeable fraction (10%) of HTTP REQs from the client are re-directected by the server to another address. The proxy intercepts the re-direct message and proactively fetches the object from the new server, before passing it on to the client
- Inefficiency 5: Persistent TCP connections
  - The proxy maintains persistent TCP connections with popular web servers, and reuses it for multiple client requests, thus avoiding the costly 3-way TCP handshake for connection setup and teardowns.
  - The proxy also maintains persistent TCP connections with the clients (as long as they are within its coverage)

Controlling Connection Parallelism



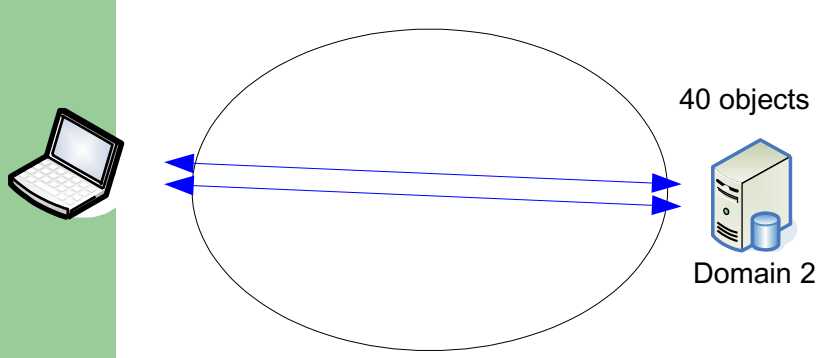
Need Algorithms to:
- Map objects to Connections
- Choose optimal number of connections

This needs to be done for Both Access and Core

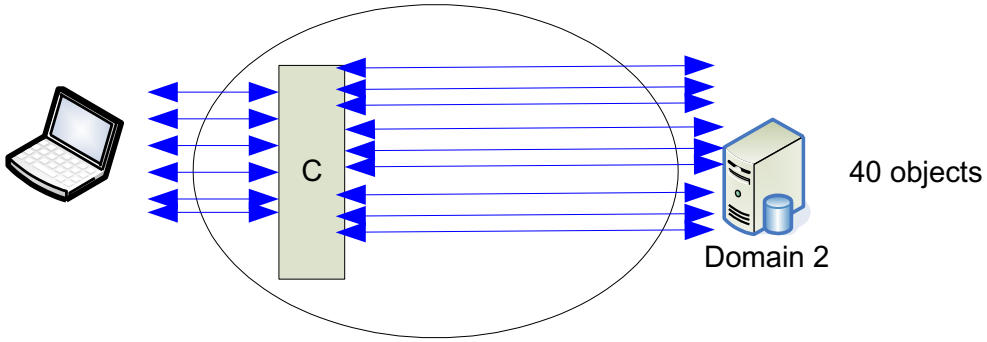
- The Proxy can control number of TCP connections to the Client
- The Proxy can control the Mapping of Individual Objects to Connections, to load balance objects across connections

- The Proxy can control the number Of TCP connections per domain. It can open more than 2 connections per domain if required
- The Proxy can use HTTP 1.1 Pipelined mode to speed up object downloads

Connection Parallelism: Example

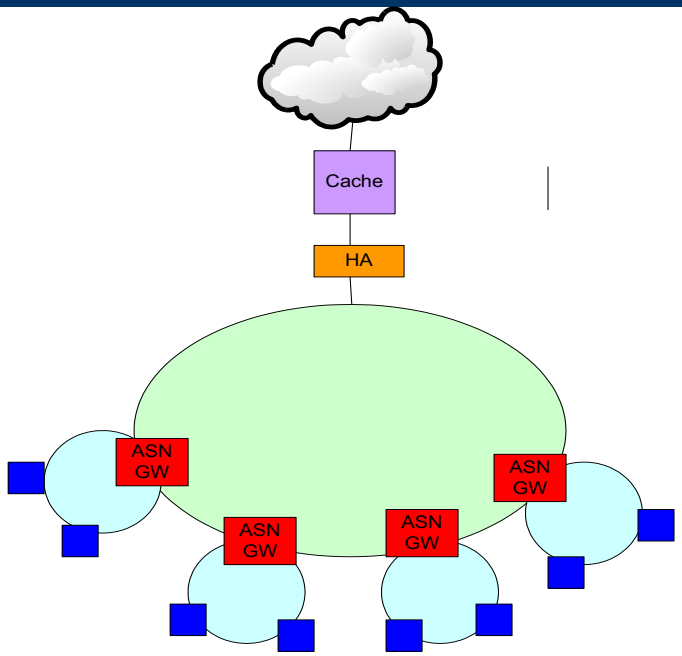


Without Proxy
- Only 2 connections to download 40 objects

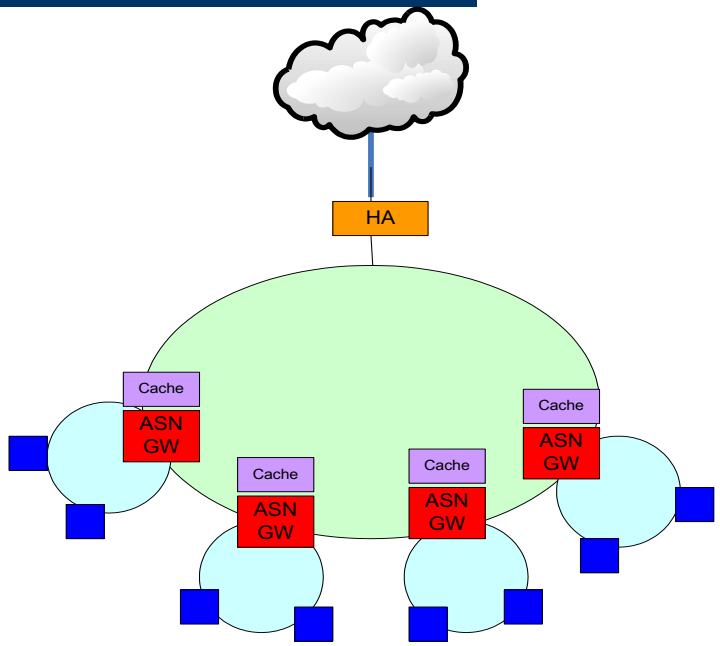


With Proxy
- Up to 6 connections to the client
- More than 6 connections to the server (With pipelining, less connections may be used)

Taking Mobility Into Account: Inter Cache Communications



Centralized Cache Architecture

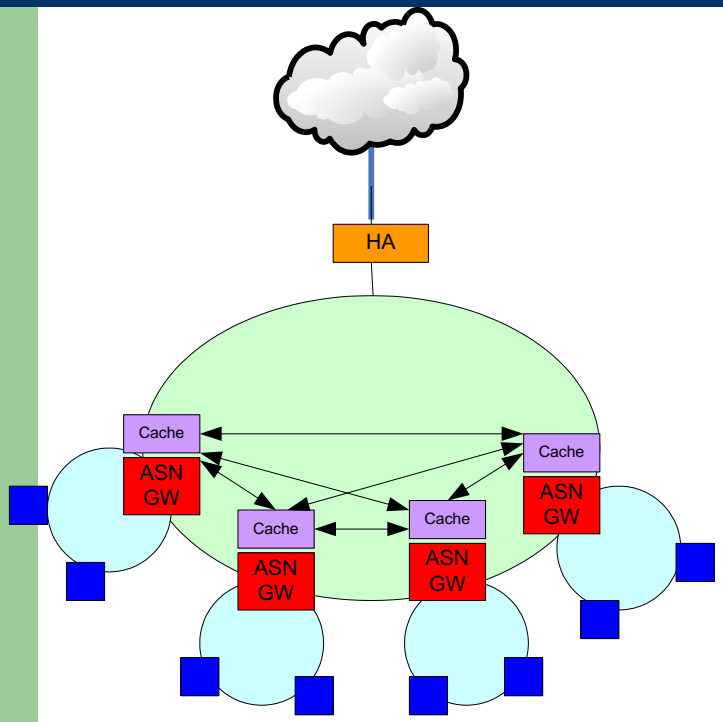


Distributed Cache Architecture

Comparison

- Centralized Proxy Cache
  - Pros
    - Higher Hit Ratio since cache covers a larger client population
- Distributed Proxy Cache
  - Pros
    - Closer to client, hence lower latency on hit
    - Saves on core network bandwidth and reduces load on Home Agent
    - Makes it possible to tailor content to suit current network and link conditions

Comparison (cont)



How can we design the Distributed Cache system so that its Hit Rate Approaches that of the centralized Cache?

By Means of Inter Cache Communication Alternatives:

- ICP (Inter Cache Communications)
- Summary Cache Protocol
- CARP (Cache Array Routing Protocol)

An Important Design Decision

- If there is a cache miss at the current ASN_GW cache, but there is a hit at one of the other caches in the group, should the object being requested be transferred to the current ASN_GW cache?
- I am currently leaning towards NOT transferring the object
  Pros of doing so:
  - Saves on cache memory, since the same object is not replicated in multiple caches. This effectively increases the total caching capacity of the system to be equal to the sum of the caches at the individual AN GWs.
- If the capacity of a single cache is sufficient to accommodate the entire client population (across all the cache in the group), then moving the object makes sense

Inter Cache Communication Protocols

- ICP
  - In case of MISS, the cache sends an UDP message to every other cache in the group. Inter
  - If the URL is found in one of the other caches, the object is sent over a HTTP connection to the requesting cache, which then can optionally store the object in its own cache. In case the object is not found in any other cache, the REQ is sent to the web server
  - The number of inter cache messages increases as O(N^2), as a result ICP is not recommended for group sizes of more than 6 or 7 caches
- CARP
  - Defines a function for mapping URLs to caches. This function has the property that if a cache is added, the items in the other caches are not affected
  - CARP is more appropriate for Cache Clusters, in which multiple cache are used together at a site in order to scale up the cache size

Summary Cache Protocol

- Basic Ideas:
  - Each cache sends periodic (or threshold based) updates of its cache directory to all other caches in group
  - The contents of the cache directory are coded using a special scheme called Bloom Filters
    - n directory entries are coded using m bits and k hash functions
    - The ratio of m/n can be varied, such that smaller ratios lead to smaller storage requirements, at the expense of higher number of false hits
- **Example**: 10 proxies, each with 8 GB of cache, each proxy stores 1 million objects (average object size 8 Kbytes), k = 4, m/n = 16 so that 2 MBytes needed for Bloom Filter memory per proxy, and 10 * 2 = 20 Mbytes for all proxies (this has to be kept in main memory).
  Size of update message = A few hundred Kbytes (only diff sent).
  Example update message trigger = When 1% of URLs change (= 10,000 entries)
  Size of update = (24 + 1) * 10,000 = 31.25 kbytes

  Corresponding number for 240 GB cache size: 30 million objects, 30 Mbytes needed for Bloom Filter memory, 300 Mbytes for other proxies, Size of update message (for 1% update threshold): 30 bits * 300K changes = 1.125 Mbytes

Handling Active Cache Traffic Streams during Handoff: Scenario 1

- Consider the following scenario: A MS does a handoff which causes its anchor ASN_GW to change from S_ASN_GW to T_ASN_GW, while it has one or more existing TCP/HTTP sessions with S_ASN_GW (due to a prior proxy cache hit)
- It is not possible to transfer the TCP sessions to T_ASN_GW, while they are still active. Hence the two ASN GWs need to maintain the R4 path for the MS for as long as these connections are open. As soon as all data has been transferred, the S_ASN_GW should do a FIN to close the TCP sessions, followed by closing of the R4 connection
- What if the MS originates a new page download after moving to T_ASN_GW, but while still connected to S_ASN_GW (due to above scenario)?
  - The T_ASN_GW transparently intercepts the REQ, and establishes a connection directly between itself and the Server to download the requested objects (or alternatively from its own or other caches in the group). Note that this data transfer does not follow the R3-R4 path between the HA and the T-ASN_GW, but flows directly between the HA and T_ASN_GW (without the use of MIP)

Handling Active Cache Traffic Streams during Handoff: Scenario 2

- Consider the following scenario: A MS does a handoff which causes its anchor ASN_GW to change from S_ASN_GW to T_ASN_GW, while it has one or more existing direct TCP/HTTP sessions with web servers or with other caches in the group
- In this case it is possible to transfer the TCP sessions to T_ASN_GW, while they are still active. Doing so involves transferring TCP state information from S_ASN_GW to T_ASN _GW. Should we do this OR should we wait until the connection is closed?
  - I am currently leaning towards the latter to simplify the design.

Cache Size Estimates

Bandwidth Savings due to Caching

- BW Savings in Core Network
- BW Savings in ASN
  - Data compression at proxy
  - Fulfillment of IMS requests for web objects (made by MS), by proxy
  - Pushing of freshness information from proxy to MS, thus avoiding the need for the MS to even send an IMS REQ
- Another compression technique that require client side processing:
  - Value Based Web Caching – This will be covered in the Compression section

SQUID Details: How does Squid decide when to refresh a cached object?

When checking the object freshness, SQUID calculate's these values:

OBJ_DATE is the time when the object was given out by the origin server. This is taken from the HTTP Date reply header.

OBJ_LASTMOD is the time when the object was last modified, given by the HTTP Last-Modified reply header.

OBJ_AGE is how much the object has aged *since* it was retrieved:

OBJ_AGE = NOW - OBJ_DATE

LM_AGE is how old the object was *when* it was retrieved:

LM_AGE = OBJ_DATE - OBJ_LASTMOD

LM_FACTOR is the ratio of *OBJ_AGE* to *LM_AGE*:

LM_FACTOR = OBJ_AGE / LM_AGE

CLIENT_MAX_AGE is the (optional) maximum object age the client will accept as taken from the HTTP/1.1 Cache-Control request header.

EXPIRES is the (optional) expiry time from the server reply headers.

These values are compared with the parameters of the *refresh_pattern* rules. The refresh parameters are:

URL regular expression

CONF_MIN: The time (in minutes) an object without an explicit expiry time should be considered fresh. The recommended value is 0, any higher values may cause dynamic applications to be erronously cached unless the application designer has taken the appropriate actions.

CONF_PERCENT: A percentage of the objects age (time since last modification age) an object without explicit exipry time will be considered fresh.

CONF_MAX: An upper limit on how long objects without an explicit expiry time will be considered fresh.

The URL regular expressions are checked in the order listed until a match is found. Then the algorithms below are applied for determining if an object is fresh or stale.

Squid Details: Refresh Algorithm

The refresh algorithm used in Squid-2 looks like this:

```
if (EXPIRES) {
                        if (EXPIRES <= NOW)
                           return STALE
                         else
                           return FRESH
                    }
if (CLIENT_MAX_AGE)
     if (OBJ_AGE > CLIENT_MAX_AGE)
         return STALE
     if (OBJ_AGE > CONF_MAX)
         return STALE
     if (OBJ_DATE > OBJ_LASTMOD)
         {
           if (LM_FACTOR < CONF_PERCENT)
              return FRESH
                 else
              return STALE
          }
if (OBJ_AGE <= CONF_MIN)
    return FRESH
return STALE
```

Squid Details: What's Cacheable

- HTTP/1.1 allows caching anything by default
  Unless overridden with **`Cache-Control`** header
- In practice, most caches avoid anything with
  - **`Cache-Control/Pragma`** header
  - **`Cookie/Set-Cookie`** header
  - **`WWW-Authenticate/Authorization`** header
  - **`POST/PUT`** method
  - **`302/307`** status code (redirects)
  - SSL content
- Study done at UC San Diego (in 2003) showed that 66% of HTTP Responses were un-cacheable, mostly due to "indiscriminate use of cookies"

Investigate and Document Algorithms used by Squid in these 2 areas

Squid Details: What's Cacheable

To determine whether a given object may be cached, Squid takes many things into consideration. The current algorithm (for Squid-2) goes something like this:

- Responses with *Cache-Control: Private* are NOT cachable.
- Responses with *Cache-Control: No-Cache* are NOT cachable.
- Responses with *Cache-Control: No-Store* are NOT cachable.
- Responses for requests with an *Authorization* header are cachable ONLY if the reponse includes *Cache-Control: Public*.
- Responses with *Vary* headers are NOT cachable because Squid does not yet support Vary features.
- The following HTTP status codes are cachable:
  - 200 OK
  - 203 Non-Authoritative Information
  - 300 Multiple Choices
  - 301 Moved Permanently
  - 410 Gone

However, if Squid receives one of these responses from a neighbor cache, it will NOT be cached if ALL of the *Date*, *Last-Modified*, and *Expires* reply headers are missing. This prevents such objects from bouncing back-and-forth between siblings forever.

A 302 Moved Temporarily response is cachable ONLY if the response also includes an *Expires* header.

Squid Details: What's Cacheable

The following HTTP status codes are "negatively cached" for a short amount of time (configurable):

- 204 No Content
- 305 Use Proxy
- 400 Bad Request
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed
- 414 Request-URI Too Large
- 500 Internal Server Error
- 501 Not Implemented
- 502 Bad Gateway
- 503 Service Unavailable
- 504 Gateway Time-out

All other HTTP status codes are NOT cachable, including:

- 206 Partial Content
- 303 See Other
- 304 Not Modified
- 401 Unauthorized
- 407 Proxy Authentication Required

Squid Details: Cache Replacement Algorithm

How does Squid's cache replacement algorithm work?

Squid uses an LRU (least recently used) algorithm to replace old cache objects. This means objects which have not been accessed for the longest time are removed first. In the source code, the StoreEntry->lastref value is updated every time an object is accessed.

Objects are not necessarily removed "on-demand." Instead, a regularly scheduled event runs to periodically remove objects. Normally this event runs every second.

Squid keeps the cache disk usage between the low and high water marks. By default the low mark is 90%, and the high mark is 95% of the total configured cache size. When the disk usage is close to the low mark, the replacement is less aggressive (fewer objects removed). When the usage is close to the high mark, the replacement is more aggressive (more objects removed).

When selecting objects for removal, Squid examines some number of objects and determines which can be removed and which cannot. A number of factors determine whether or not any given object can be removed. If the object is currently being requested, or retrieved from an upstream site, it will not be removed. If the object is "negatively-cached" it will be removed. If the object has a private cache key, it will be removed (there would be no reason to keep it -- because the key is private, it can never be "found" by subsequent requests). Finally, if the time since last access is greater than the LRU threshold, the object is removed.

The LRU threshold value is dynamically calculated based on the current cache size and the low and high marks. The LRU threshold scaled exponentially between the high and low water marks. When the store swap size is near the low water mark, the LRU threshold is large. When the store swap size is near the high water mark, the LRU threshold is small. The threshold automatically adjusts to the rate of incoming requests. In fact, when your cache size has stabilized, the LRU threshold represents how long it takes to fill (or fully replace) your cache at the current request rate. Typical values for the LRU threshold are 1 to 10 days.

Back to selecting objects for removal. Obviously it is not possible to check every object in the cache every time we need to remove some of them. We can only check a small subset each time.

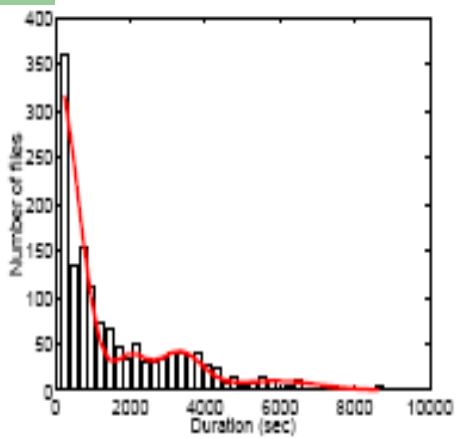
Every time an object is accessed, it gets moved to the top of a list. Over time, the least used objects migrate to the bottom of the list. When looking for objects to remove, we only need to check the last 100 or so objects in the list. Unfortunately this approach increases our memory usage because of the need to store three additional pointers per cache object. We also use cache keys with MD5 hashes.

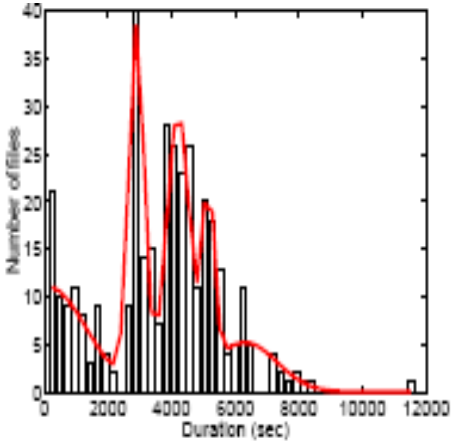
Streaming Media Caching

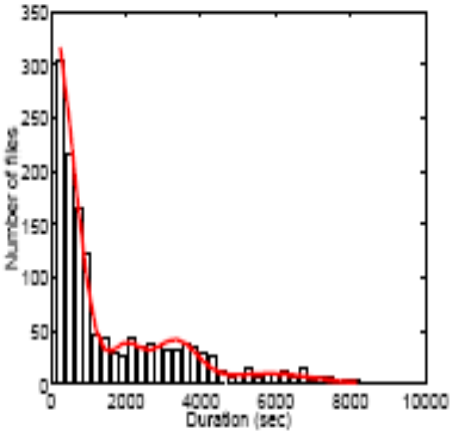
Streaming Media Caching

- Requirements
  - The design should support an RTSP proxy, to intercept client RTSP requests to the server, and act on them locally. The design should support the following streaming media engines (that use RTSP): Apple Darwin, Real Networks Helix (server and proxy),
  - The design should support an Windows Media Technology (WMT) proxy, to support Microsoft MMS (Microsoft Media Server)
  - The design should support a proxy for Flash based video streams
  - The design should support splitting of live streams and multicasting
  - Both transparent as well as non-transparent (manual) proxy modes should be supported
  - The design should make sure that all streams served from the cache are fresh
  - The design should support variable bit rate video, I.e., it should adjust the bit rate depending upon the device capabilities (using layered coding and transmission)
  - The design should be able to adjust the bit rate depending upon the current wireless link conditions
  - The design should support caching of partial media streams in order to reduce caching memory size. Alternative techniques: Sliding Interval Caching, Prefix Caching, Segment Caching, Rate Split Caching
  - The design should support Dynamic Re Location of Cache content (on a per user basis), in response to client mobility

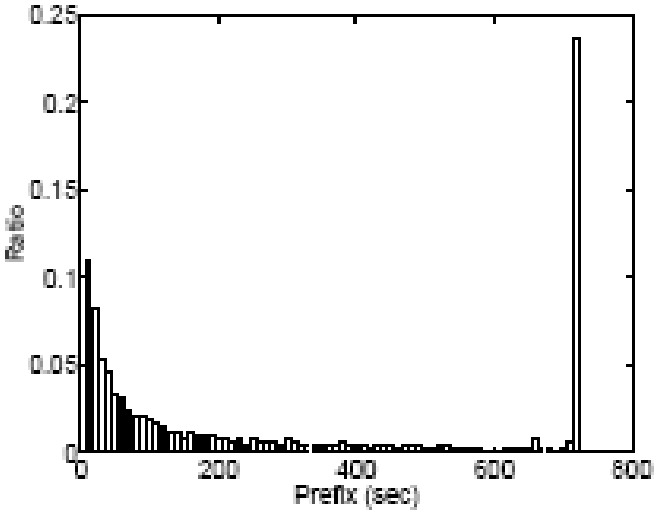
Video Traffic Properties: Duration of Session (Size of Object)

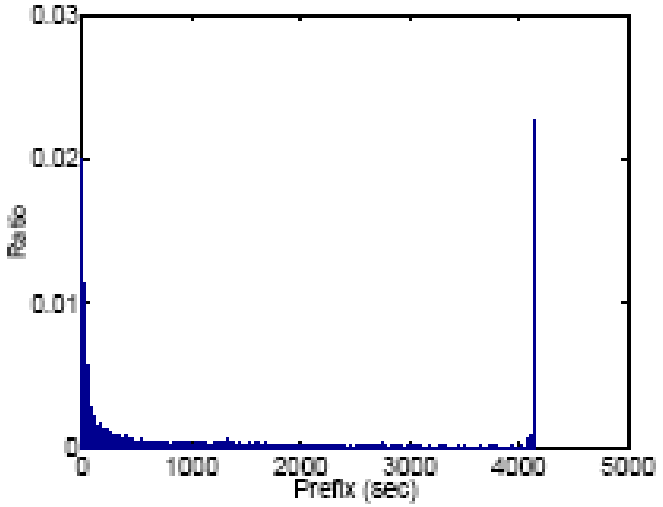






Video Traffic Properties: Incomplete Sessions





Video Traffic Properties: Object Popularity

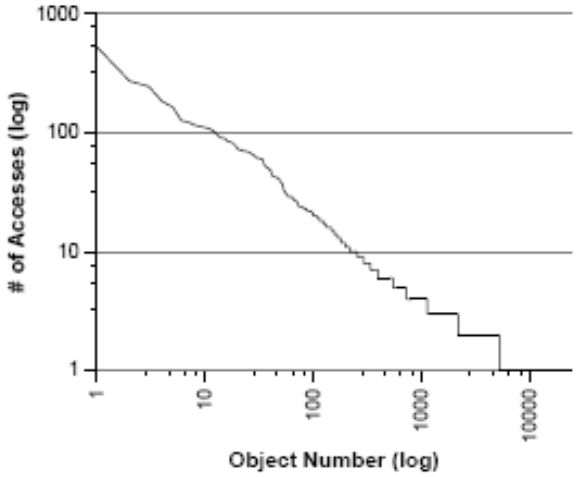


Figure 8: *Object popularity by number of sessions (note log scale).*

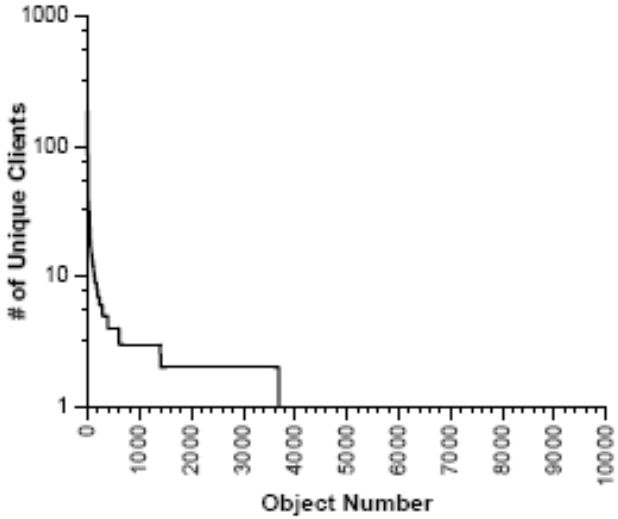
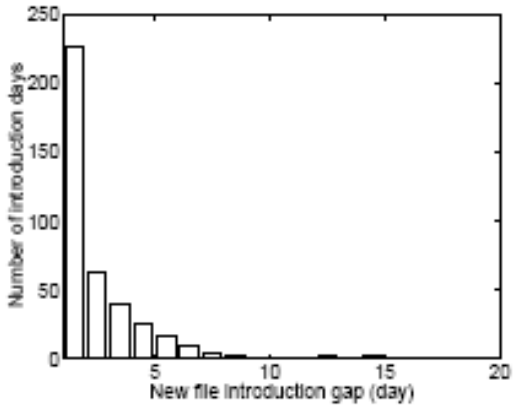


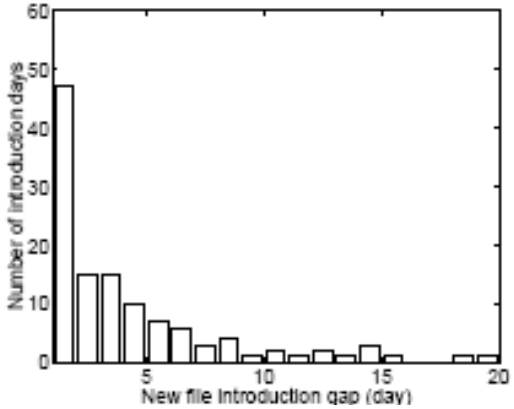
Figure 10: *Object sharing.*

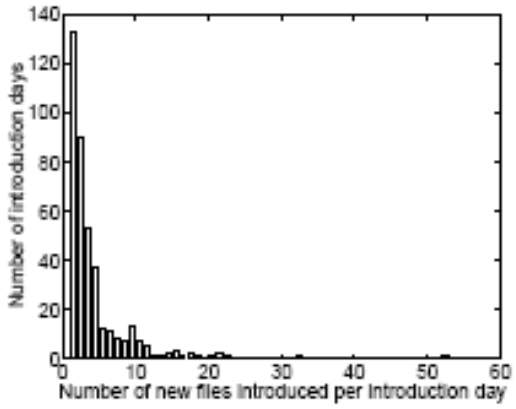
- Total number of media objects = 23,738
- 78% (18,515) were accessed only once
- Only 1% of objects (237) were accessed by 10 or more sessions
- The 12 most popular objects were accessed more than 100 times each
- Only 16% of objects (3800) were shared (accessed by 2 or more clients), yet requests for these shared objects account for 40% of all sessions recorded

Policies to limit size of video cache:
- Cache object only on second hit
- Aggressive cache removal policy

Video Traffic Properties: Traffic Driven by New File Introduction Process







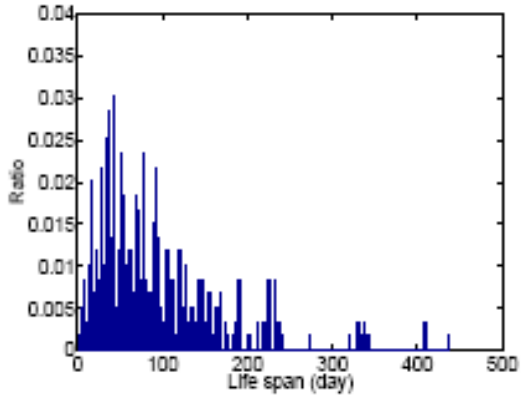


Figure 21: A *regular* lifespan.

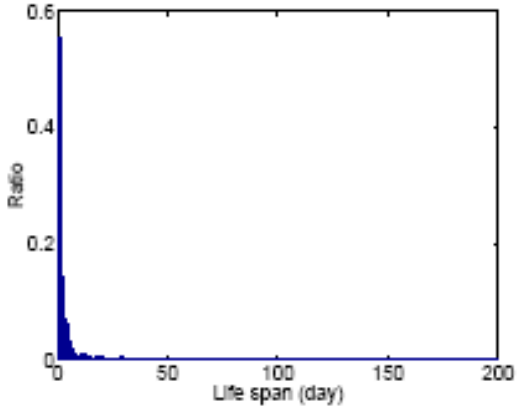


Figure 22: A *news-like* lifespan.

Video Traffic Properties: The Flash Crowd Phenomena

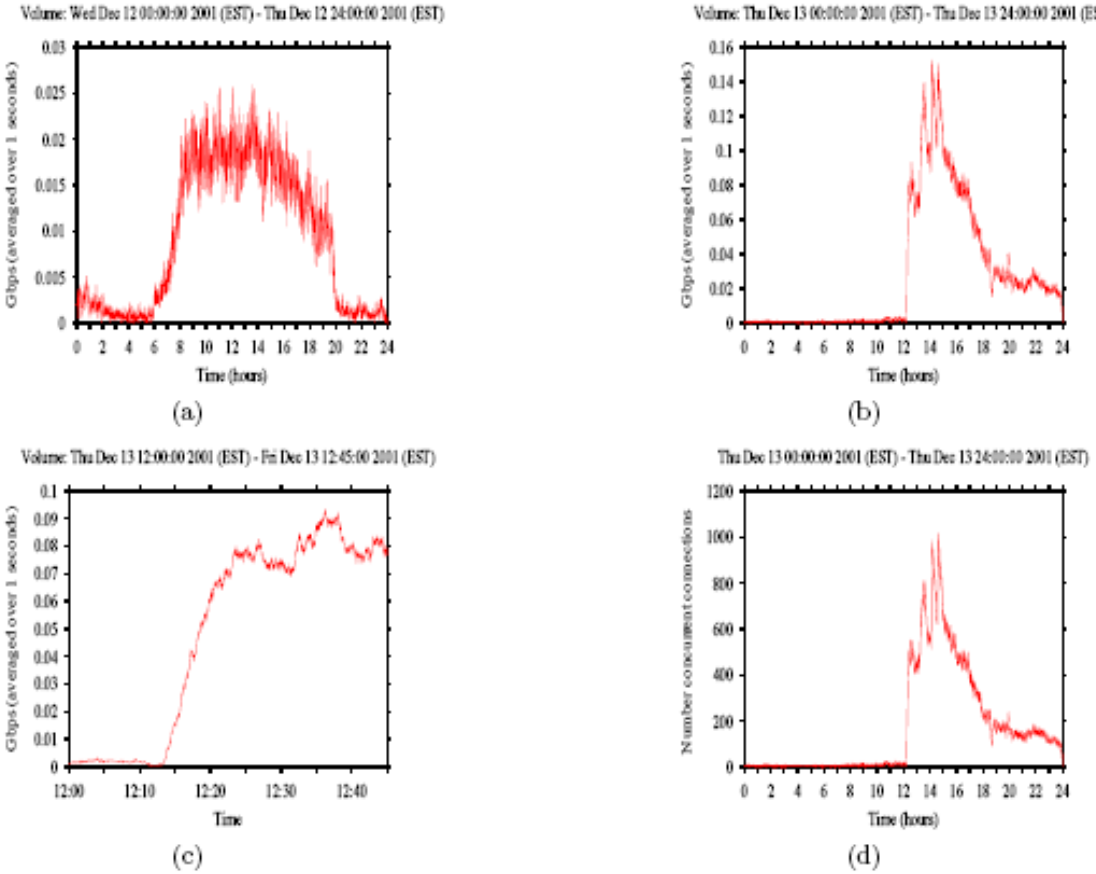


Figure 7: On Demand - (a)-(b) plot the bandwidth across time for Dec 12 and Dec 13, 2001. For the latter day, (c) plot the bandwidth for a 50 min. time interval, and (d) number of concurrent connections for the entire day.

Video Traffic Properties: IPR Issues

- Copy Protection: Rogue proxies may make un-authorized copies of the media content. Does there exist a mechanism between the origin server and proxies such that only authorized proxies can cache the content?
- Authorization: A method for origin servers to authorize caching proxies to server content from cache (per viewing basis or on a periodic basis)
- Access Accounting: A robust mechanism for the origin servers to know about hit counts and hit durations from the caching proxies

Streaming with Partial Caching

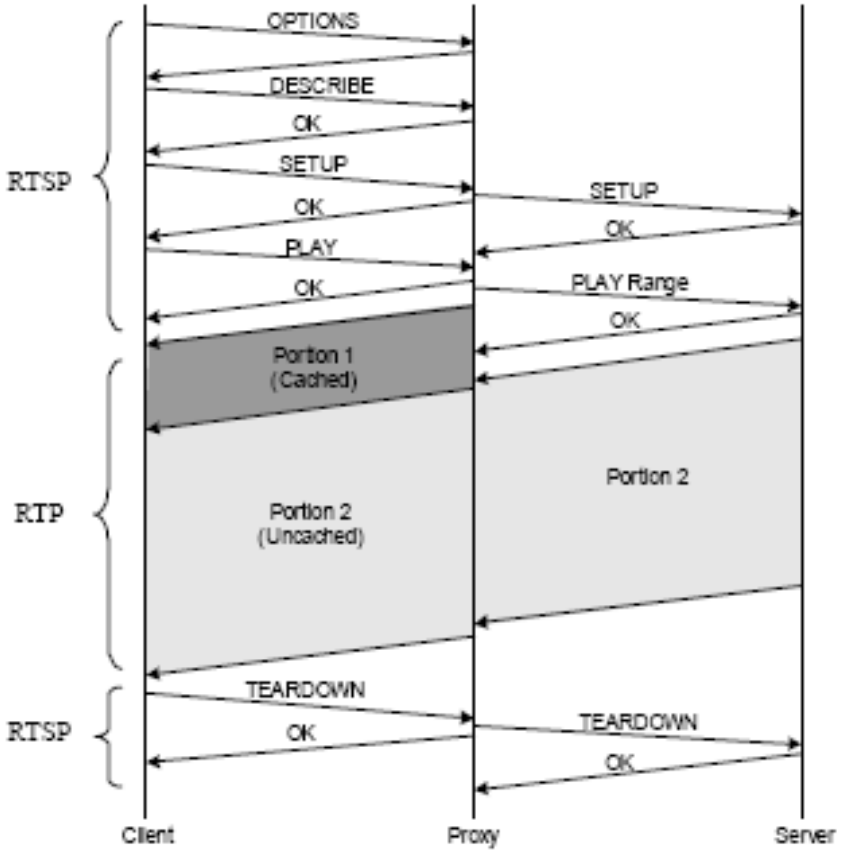
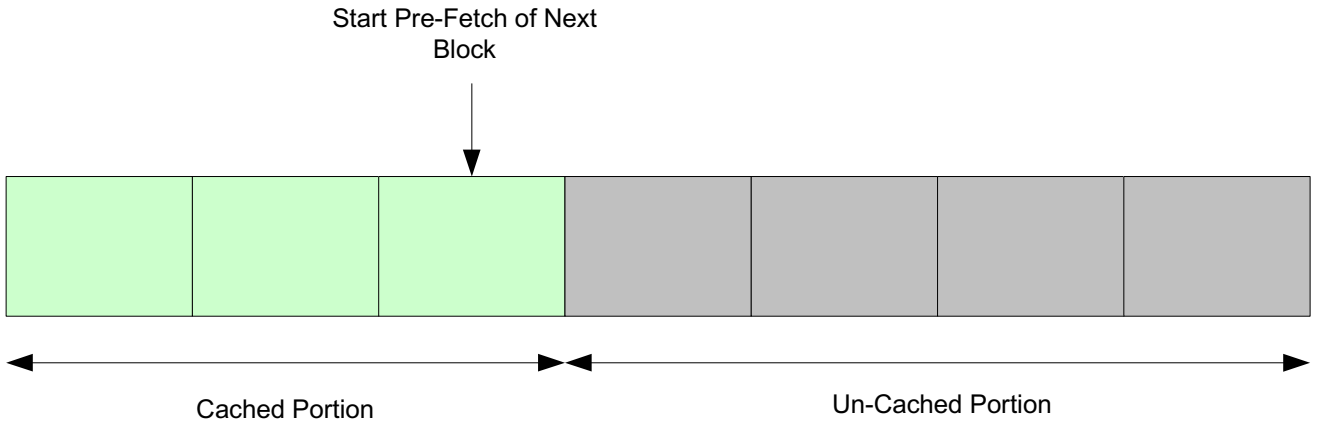


Figure 1.2. Operations for streaming with partial caching.

Proposed Caching Algorithm

- Store streaming media objects (SMO) in only one proxy cache, among a group of proxy caches
  - **Justification**: The size of a typical SMO is about 1000 to 10,000 times that of a typical web object (5 – 10 Mbytes for audio, 10 – 1000 Mbytes for video vs 4 – 10 Kbytes for typical web objects)
- The ICP algorithm, rather than the Summary Cache Algorithm, is used to check whether a SMO is in one of the sibling caches
  - **Justification**: The number of SMOs will be much smaller than the number of web objects, hence the extra traffic created by ICP will be much less. Also with ICP caches can get up to date information about the state of the other caches vs SCP were the directory is updated on a periodic basis
- Cache a SMO only when it gets accessed a second time (by any of the MSs served by the group of caches)
  - **Justification**: More than 70% of SMOs are accessed only once, hence by doing so we automatically exclude them from being cached

Caching Algorithm (cont)



- Each SMO is divided into fixed size segments (for example 200 Kbytes per segment). If N segments of a SMO already stored in a cache, the (N+1)rst segment is pre-fetched after half of the Nth segment has been streamed to the MS
  - **Justification**: A large percentage (>50%) of sessions are in-complete

Caching Algorithm (Cont)

- Distributed Hit Accounting
  - On first hit, the target proxy starts to advertise availability of the object, even though it did not store. On second hit (by possibly another MS at some other proxy), the serving proxy downloads object from target proxy
- Distributed Load Balancing
  - Each cache sends a periodic update of the current utilization of its SMO cache space
  - When a cache whose SMO cache utilization exceeds a certain threshold receives a new RTSP request which cannot be satisfied from any of the caches in the group, it forwards it to a sibling cache whose utilization is the least within the group. The target cache then may cache the object if it is requested more than once

Cache Replacement Algorithm

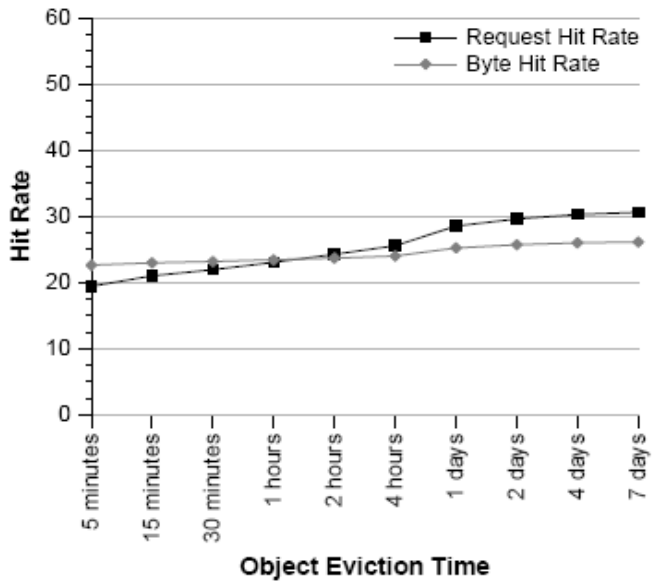
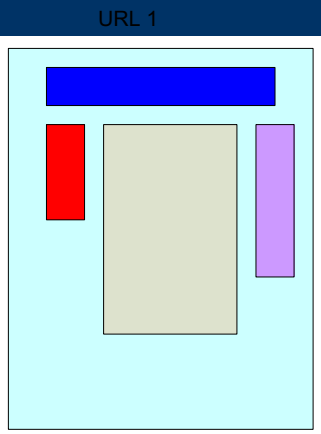


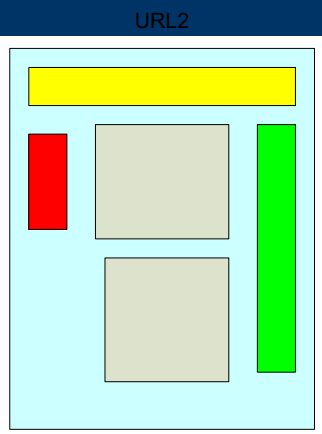
Figure 15: *Effect of eviction time on cache hit rates.*

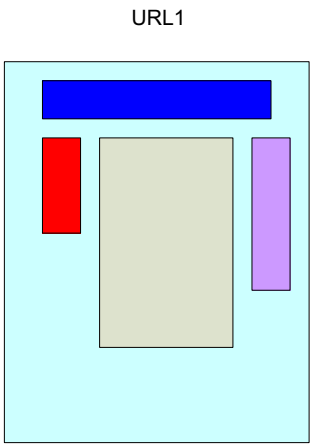
- Evict an object from the cache if the time since it was last accessed exceeds a threshold (independent of the actual cache memory utilization)
- Investigate use of LRU if the cache utilization exceeds some threshold

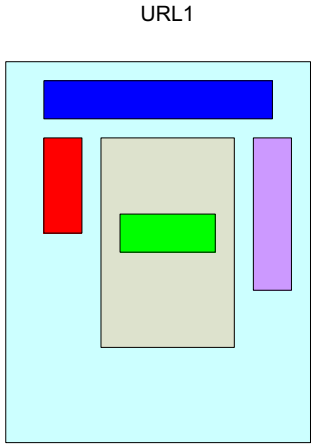
Lossless Compression

Hash Based Compression (HBC): Motivation









Redundancy in Space

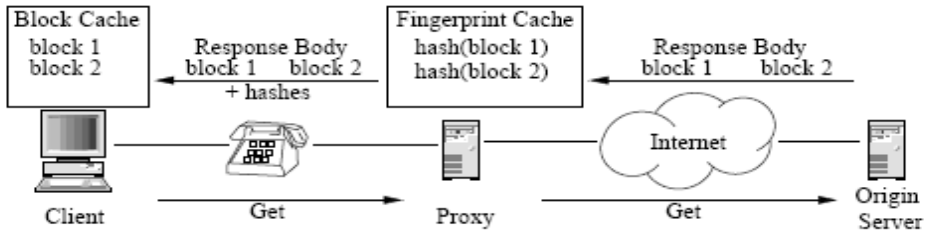
- The same object appears In different URLs: This Is called Aliasing
- Mogul et al (2002): Upto 54% of all Web transactions and 36% of all bytes transferred involve Aliased payload

Redundancy in Time

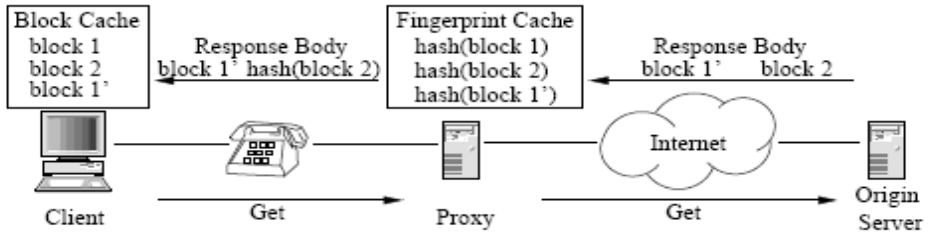
- An already cached object Changes, but only in certain Parts

HBC (cont)

First request:



Subsequent request:



Benefits:

- Reduces BW consumption due to both aliasing and resource modification
- Works even for "non-cacheable" content, and hence is effective even for personalized or dynamic web content

Both these sources of redundancy can be exploited to reduce the BW consumption between the proxy and the MS by doing the following:

- Index cached web objects by Name (URI) as well as by its value
- Break up a web object into blocks (for example 2 Kbytes each), and assign it a value using a hash function (for example MD5)
- In case of a miss transfer both the block as well as the hash to the client. **The block can be compressed further using gzip**
- In case of a hit, transfer only the hash to the client

HBC Implementation

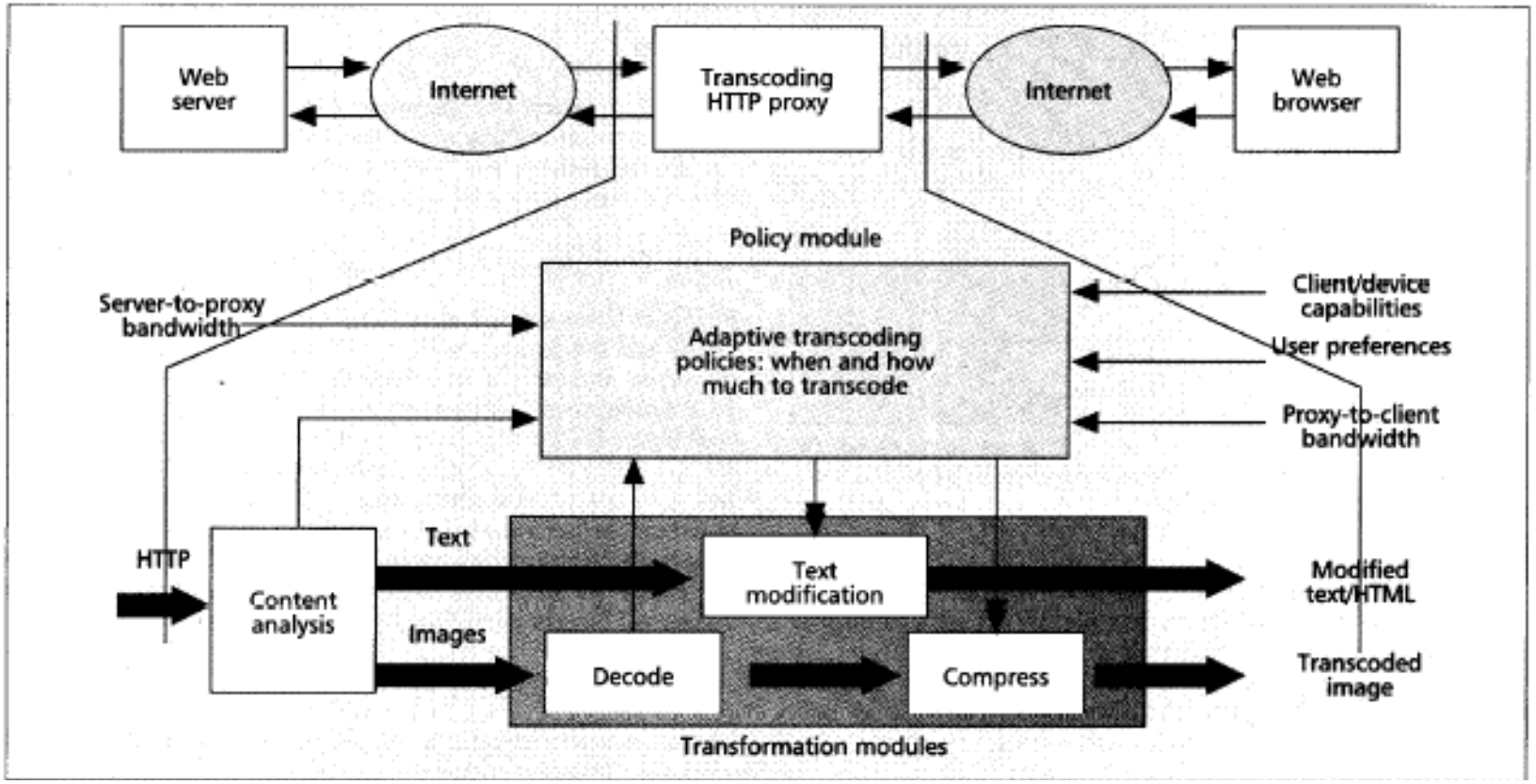
- In order to realize BW savings over the backhaul as well as the airlink, the client side of the HBC should be implemented in the MS
- The proxy and the client caches need to be synchronized in order for HBC to work, I.e, the proxy should have a current view of all the blocks in the client cache
- In the presence of mobility, the client cache directory needs to be transferred over to the new proxy
  - This can be done using the same technique as used in the Summary Cache Protocol, and can be done as part of the Handoff procedure

GZIP

- Fixed fidelity (lossless) compression: Applied to HTML, CSS, JS etc. Uses gzip – Can be used in concert with HBC

Dynamic Content Adaptation for Images

Overall Architecture for Image Transcoding



GIF/PNG vs JPG Comparison (from Wikipedia)

Comparison with JPEG

Composite image demonstrating difference between JPG and PNG.

JPEG will produce a smaller file than PNG for photographic (and photo-like) images since it uses a lossy encoding method specifically designed for photographic image data. **Using PNG instead of a high-quality JPEG for such images would result in a large increase in filesize (often 5-10 times) with negligible gain in quality.**

PNG is a better choice than JPEG for storing images that contain text, line art, or other images with sharp transitions that do not transform well into the frequency domain. Where an image contains both sharp transitions and photographic parts a choice must be made between the large but sharp PNG and a small JPEG with artifacts around sharp transitions.

JPEG is a poor choice for storing images that require further editing as it suffers from generation loss, whereas lossless formats do not. This makes PNG useful for saving temporary photographs that require successive editing. When the photograph is ready to be distributed, it can then be saved as a JPEG, and this limits the information loss to just one generation. That said, PNG does not support Exif image data from sources such as digital cameras, which makes it problematic for use amongst amateur and especially professional photographers. TIFF does support it as a lossless format, but is much larger in file size for an equivalent image.

JPEG has historically been the format of choice for exporting images containing gradients, as it could handle the color depth much better than the GIF format. However, any compression by the JPEG would cause the gradient to become blurry, but a 24-bit PNG export of a gradient image often comes out identical to the source vector image, and at a small file size. As such, the PNG format is the optimal choice for exporting small, repeating gradients for web usage

GIF → JPEG Transformations

- GIFs should not be used for color photographs: How can the system identify such images and transform them to JPEG?
- GIF Classification scheme from Chandra et al:
  - Bullets: Images smaller than 25 X 25 pixels
  - Lines: Images whose width exceeds 300 pixels and height less than 25 pixels
  - Icons: Images between 25 X 25 and 100 X 100 pixels
  - Banners: Images whose width exceeds 300 pixels and height between 25 and 100 pixels
  - TrueImages: Those that do not belong to any of these categories
  - Large True Images: True Images whose size exceeds 40 KB

Only Large GIF TrueImages should be transformed into JPEGs

Quality Factor: A JPEG Compression Metric

JPEG Compression Steps

1. Convert Color Space
2. Downsample
3. Forward DCT
4. Quantize
5. Entropy Encode

$$S = (Q>=50)?(5000/Q) : (200 - 2Q)$$
$$imgQuantTbl[I] = (stdQuantTbl[I]*S+50)/100$$

S: Scaling Factor
stdQuantTbl: Standard Quantization Table

- We need a metric that corresponds to the user's perception of quality of a JPEG image
- Transcoding efficiency: Ability of a transcoding algorithm to loose more in storage space for a particular loss in information quality
- In JPEG, the compression ratio of the output image is controlled solely by the quantization tables (used during the quantization step of the compression)
- Quantization tables can be scaled to vary the compression ratios. Most JPEG compressiors allow the user to specify a range of values for the scaling factor, by specifying a metric called the Quality Factor
- Transcoding that uses JPEG Quality Factor can transcode an image, either to an absolute Quality Factor value or to a percentage of the original quality

Compression Metric (cont)

- The Quality Factor for an image can be computed using the following formulae:

  S = imgQuantTbl[I] * 100 – 50/stdQuantTbl[I]

  Q = (S <= 100)?(200 – S)/2 : (5000/S)

- Efficient Transcoding: If an image is transcoded to loose 50% of information quality, the output image size should be less than 50% of the original image size
- Experimental Finding: Images reduce efficiently for smaller drops in quality

  75% Drop in Quality → 66.1% of the images transcoded efficiently

  50% Drop in Quality → 47.4% of the images transcoded efficiently

  25% Drop in Quality → 3.2% of the images transcoded efficiently

Link Aware Image Transcoding for JPEG

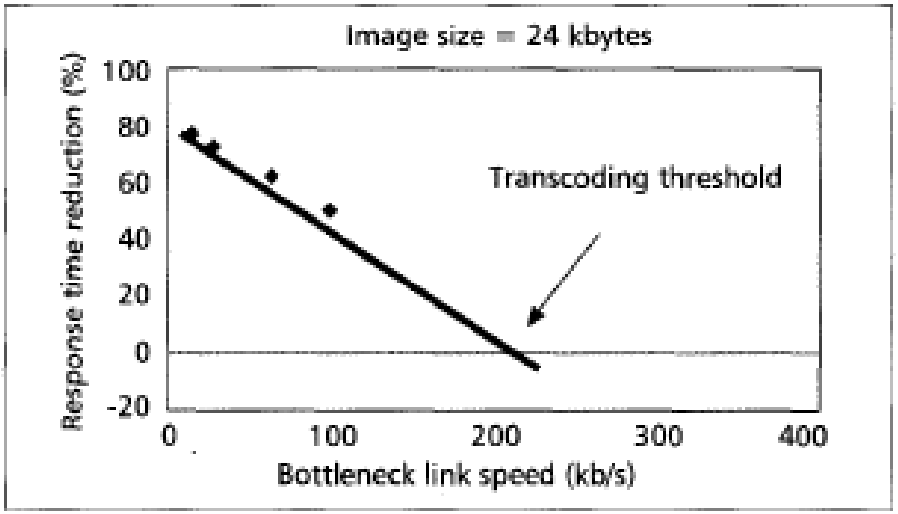


Figure 4. *Response time is reduced, and transcoding should be performed, only when the bottleneck bandwidth is below an image-dependent threshold.*

- The TCP algorithm on the hop between the proxy and the MS provides estimates of the bottleneck BW between these two devices
- The proxy estimates the time required to download the image (D = Image Size/Bottleneck BW). If D > 5 sec, then the image is transcoded to a smaller size (= 20 Kbytes)

JPEG Image Transcoding to Reduce Backhaul and Wireless Link BW Consumption/User Preferences

Example Policies:

1. User Control:
   Allow the user to set the amount of image size reduction that he desires (as a percentage of the original image size)
2. Operator Control:
   As a function of the backhaul link utilization,
   - Transcode images that are less than 40 KB to 50% of the initial JPEG Quality factor
   - For images that are larger than 40 KB, aggresively transcode them to a target size of 20 KB

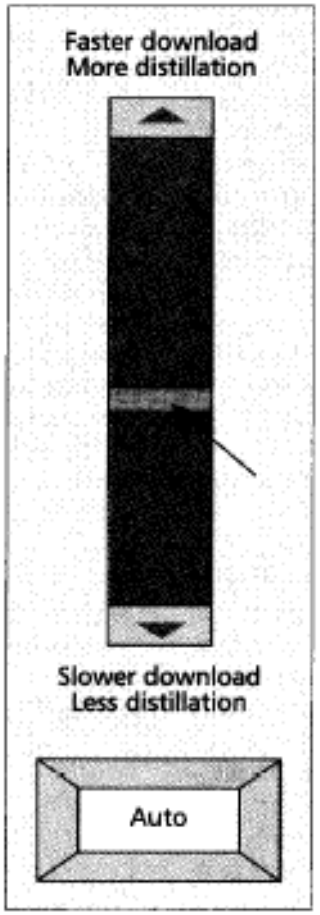
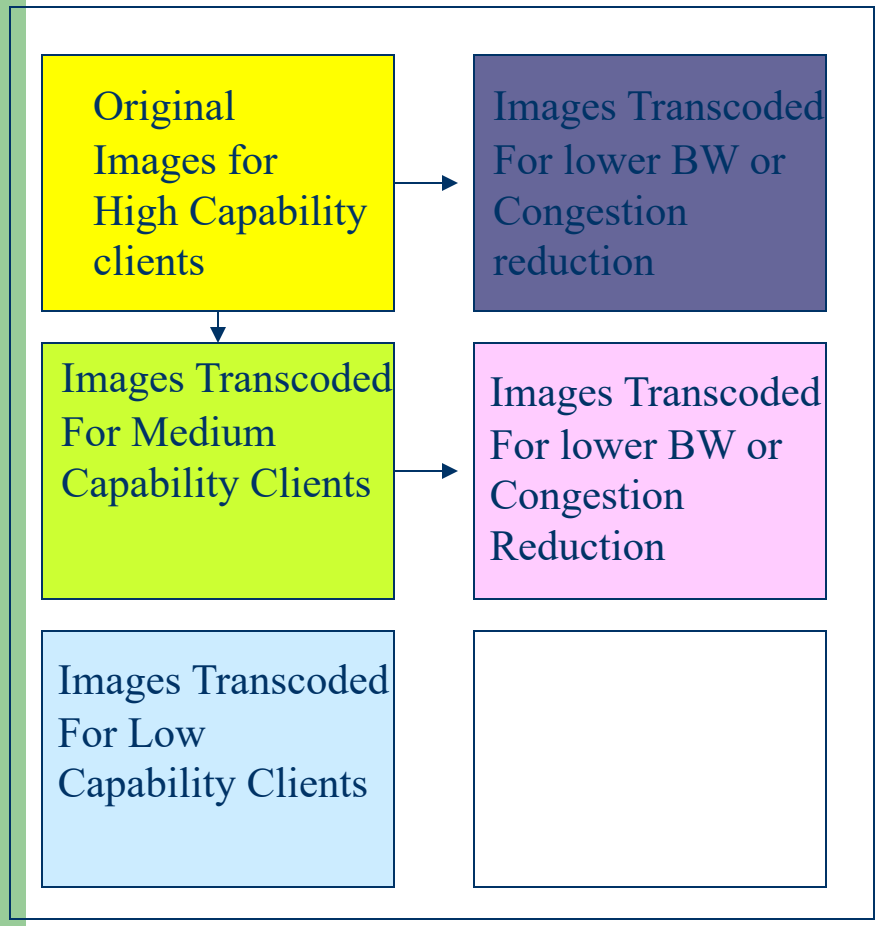


Figure 3. *User preferences in terms of quality and response time are specified using a slide bar and then communicated to the proxy.*

User Device Aware Transcoding

- Use the CC/PC protocol to discover user device capabilities, and then transcode accordingly. For example:
  1. High Capabilty Clients, such as PCs, laptops etc
     - Use transcoding to reduce BW consumption and/or to reduce link congestion
  2. Medium Capability Clients, such as smart phones
     Transcode to lower screen size (typically 320 X 200), limited colors
  3. Limited Capability Clients, such as regular cell phones
     These typically use WAP

Caching of Transcoded Images



Examples of Caching Policies:

1. Cache only originals, do all transcoding on the fly
2. Cache originals + images for client capabilities, do BW/congestion transcoding on the fly
3. Cache all levels of images

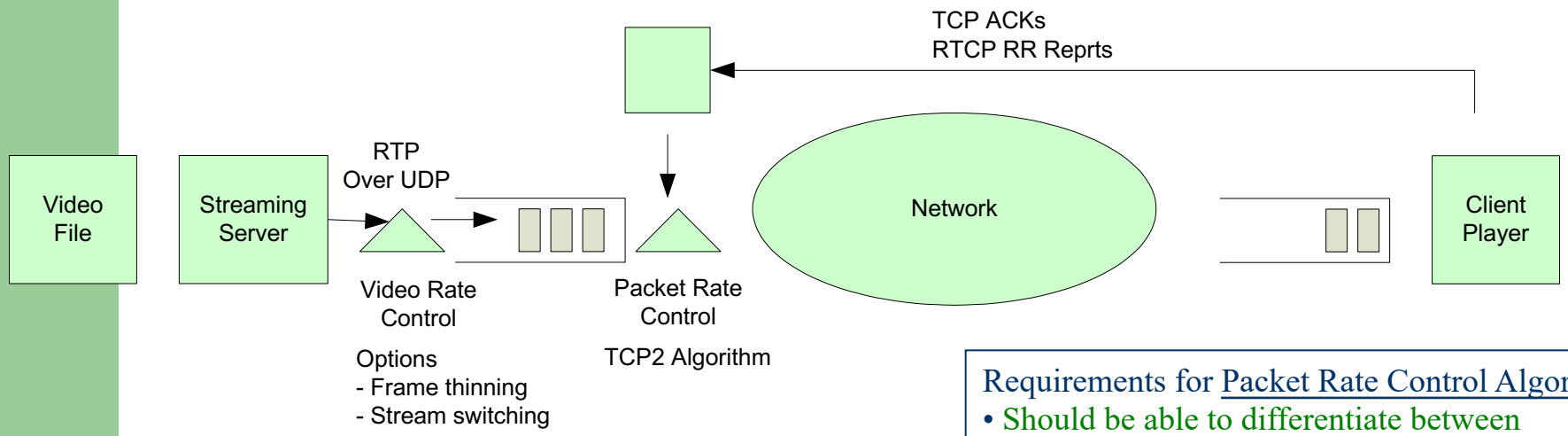
Projects that have Modified SQUID for Storing Transcoded Images

1. TranSquid: IIT, Bombay and University if Massachusetts
2. Soft Caching Proxy Project: USC
3. Distributed Proxy Caching for Transcoded Images: University of Rome

Dynamic Rate Control and Content Adaptation for Video

Traffic Control Framework for Transport of Video over RTP over TCP

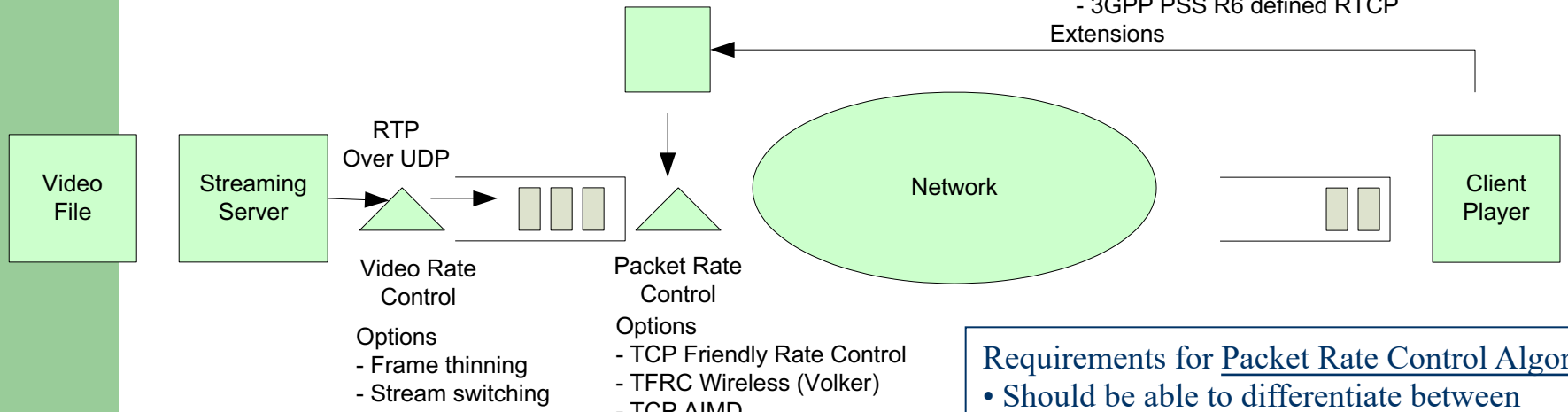
Appropriate for Streaming Stored Video With a few seconds of Buffering at the Client



Requirements for Video Rate Control
- Should be simple to implement and should work for both stored as well as live streams
- Should reduce the video rate to a value that is lower than the Packet Rate

Requirements for Packet Rate Control Algorithm:
- Should be able to differentiate between congestion loss, link error and handoff loss
- Should use only standards based feedback mechanisms
- Should not change packet rate very often (smooth rate variation), while sending the best quality video that the conditions will allow
- Should prevent overflow and underflow of Playout buffer

Traffic Control Framework for Transport of Video over RTP over UDP



Requirements for Video Rate Control
- Should be simple to implement and should work for both stored as well as live streams
- Should reduce the video rate to a value that is lower than the Packet Rate

Requirements for Packet Rate Control Algorithm:
- Should be able to differentiate between congestion loss, link error and handoff loss
- Should use only standards based feedback mechanisms
- Should not change packet rate very often (smooth rate variation), while sending the best quality video that the conditions will allow
- Should prevent overflow and underflow of Playout buffer

Elements of Rate Control Algorithm

- Rate control changes should happen in response to changes in airlink bandwidth, and NOT in response to air-link losses
  - This objective can be achieved by eliminating congestion losses, so that all losses are due to the air-link
  - TFRC cannot be used since TFRC changes rate in response to congestion losses
- Receiver Feedback will be by means of RTCP Receive Reports (RR)
- Slow Start will not be used, the initial rate corresponds to the intrinsic rate of the video stream
- Rate Increase Rule: Rate will be increased by at most 1 extra packet every RTT
- Rate Decrease Rule: If the Network Backlog estimate exceeds a threshold, then the rate should be decreased to the Smoothed Receive Rate

RTCP Receive Reports – RFC 3550

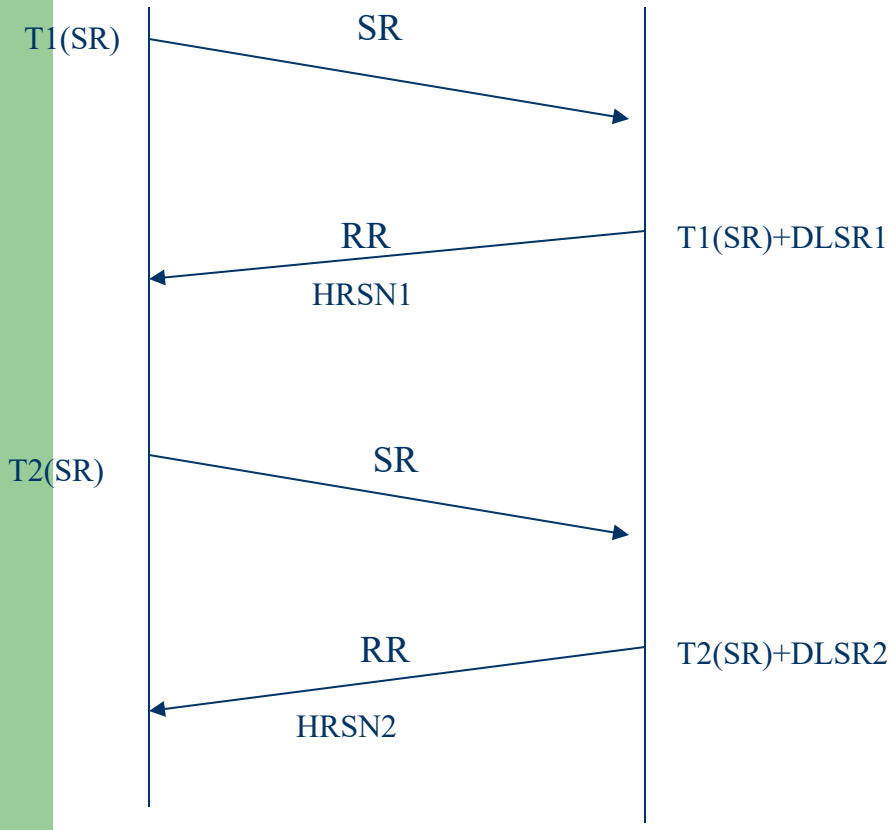
- Should be sent at least once every RTT
- Information in a Standard RR
  1. Fraction Lost: Fraction of RTP data packets lost since previous RR packet was sent
  2. Cumulative Number of Packets Lost: Number of packets lost since beginning of reception
  3. Highest Received Sequence Number (HRSN)
  4. Inter-arrival Jitter
  5. Last SR Timestamp (LSR)
  6. Delay Since Last SR (DLSR): Along with (5), can be used to calculate the RTT = A – LSR – DLSR, where A is the time at which the sender gets the RR.
- NACK Information in RFC 4585 (AVPF) defined RR
- RTCP/RTSP Extensions in 3GPP R6 PSS

How widely supported are these?

RTCP/RTSP Extensions in 3GPP R6 PSS Specification

- New fields in the RTCP RR packet:
  - Playout Delay
  - NSN: The RTP Sequence Number of the next ADU to be decoded
  - NUN: The unit number (within RTP packet of the next ADU to be decoded
  - FBS: The amount of Free Buffer Space available in the client (in increments of 64 byte blocks)
- RTSP Extensions
  - Link Characteristics Header: Guaranteed BW, Max BW and Max Transfer Delay, across the wireless link
  - Adaptation Header:
    - Buffer Size: Reception, de-jittering and possibly de-interleaving buffer
    - Target Protection Time: Minimum amount of playback time (in ms) that the client perceives necessary for interrupt free playback

Receive Rate and Network Backlog Computation from RTCP Reports



$$NB = PS(HTSN) + \ldots + PS(HRSN)$$

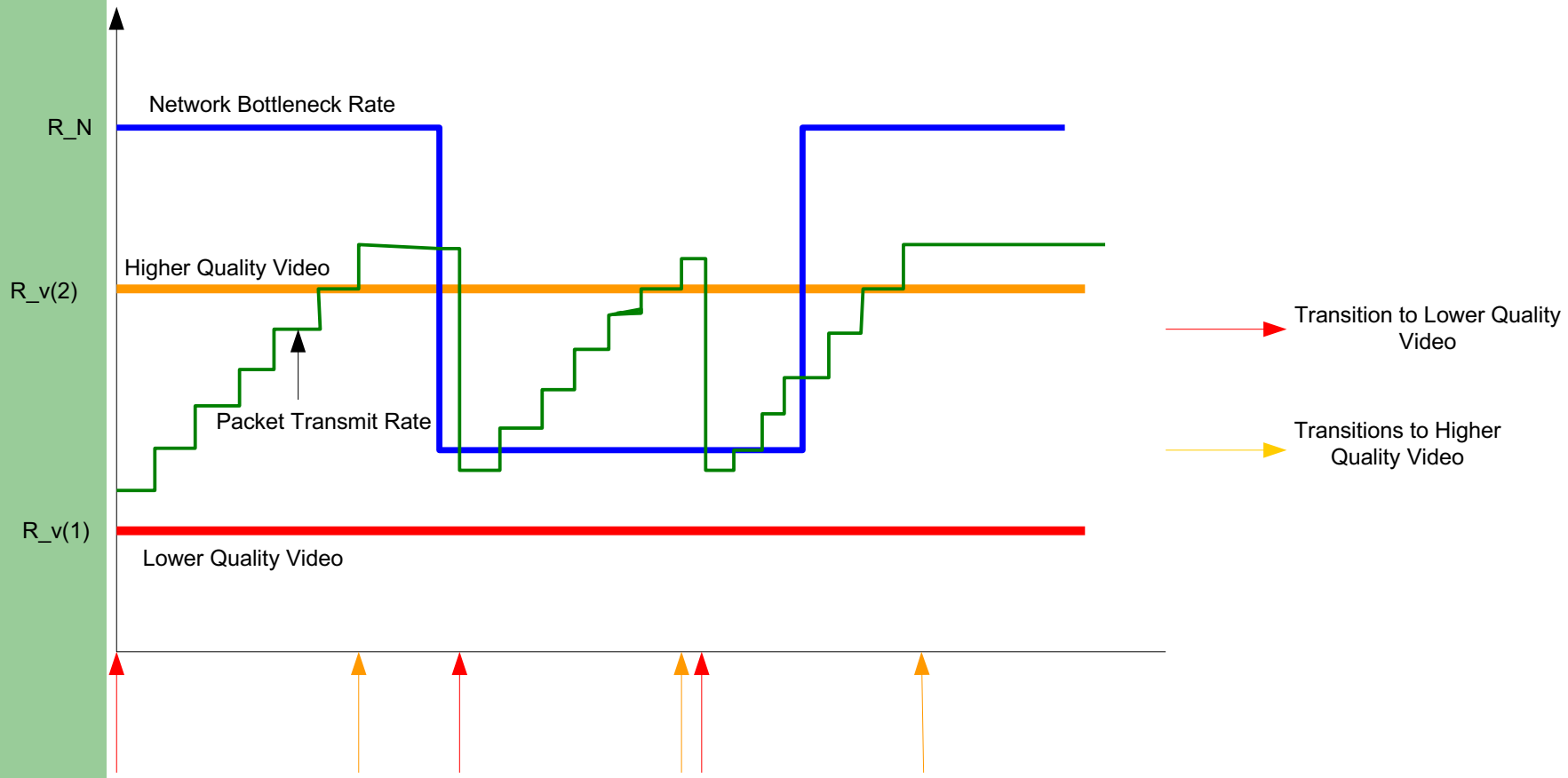
$$B = \frac{[PS(HRSN1) + \ldots + PS(HRSN2)]}{T2(SR)+DLSR2-T1(SR)-DLSR1}$$

$$C(I+1) = a*C(I) + (1-a)*(B(I+1)+B(I))/2$$

Packet and Video Rate Control Rules

- Initial Rate = Full Rate of Video Stream
- Rate Decrease Rule
  - Assuming Basic RTCP RR
    Decrease Rate if, NB > Threshold
    Set New Rate R = C
    Set Video Rate to R_v, such that R_v < C
  - Assuming PSS enhanced RTCP RR
    Decrease Rate if, NB > Threshold AND Remaining Playout Time < Threshold
    Set New Rate R = C
- Rate Increase Rule
  - If R > R_v(max)
    Do not increase R
    else
    R → (R + 1/RTT)/(2 – RTT'/RTT)
    if R > R_v(next)
    R_v = R_v(next)

Video Rate Control Example



Handling Handoffs and Link Errors

- Handoffs
  - If the ASN Data Handoff Integrity protocol is present, then it can be used to do do buffering and/or bicasting during handoffs. This can potentially make the handoff transparent to the video traffic
  - If this protocol is not present, then handoffs can be handled using a combination of Layer 2 handoff hints and the AVPF RTCP feedback from the receiver
- Link Errors
  - If the re-transmission can be done without violating the playout constraint, then the AVPF RTCP feedback reports can be used to identify lost packets by the receiver

Frame Thinning

Apple QuickTime

Supports the following protocols:

- RTSP over TCP
- RTP over UDP
- RTP over Apple's Reliable UDP: Used if the client requests it. A set of QoS enhancements including: Congestion Control, Tuning Improvements, Retransmits, Thinning Algorithms
- RTSP/RTP in HTTP
  - Enables clients to access Quicktime files while located behind HTTP proxies
- RTP over TCP (RFC 2326)

Apple Reliable UDP Features

- Client acknowledgement of packets sent by the server to the client
  - Server expects to receive an ACK for each RTP packet
  - Bit mask based ACK packet format used
- Windowing and congestion control so the server does not exceed the currently available BW
- Server retransmissions to the client in the event of packet loss
- Faster than real time streaming known as “overbuffering”

Apple: RTP Payload Meta Information

The Server may supply the following additional RTP Payload Meta information to the Caching Proxy:

- Transmission Time: Recommended transmission time of the RTP packet (in ms), offset from the start of the media presentation
- Frame Type: Can be used by the Proxy to distinguish between I, B and P frames. Useful for doing Frame Thinning
- Packet Number: Value is the packet number offset from the absolute start of the stream
- Packet Position: Byte offset of this packet from the absolute start of the stream

Are these Apple proprietary RTP extensions or can they also work over other video servers ??

Microsoft Window Media Services

Fast Streaming, consists of the following 4 components:

- **Fast Start**: Provides an instant-on playback experience with no buffering delay – Provides data for pre-buffering directly to the buffer at speeds higher than the bit rate of the content requested. After the initial buffer requirement is fulfilled, the server streams at the bit rate defined by the content stream
- **Fast Cache**: Applies only for on demand streams. Streaming of content to the player cache as fast as the network will allow, for example a 128 kbps stream is transmitted at 700 kbps and buffered. Provides better user experience by making the connection more tolerant to network bandwidth fluctuations
- **Fast Recovery**: Provides redundant packets of information to clients that are using wireless connections, enables player to recover lost or damaged data packets without requesting re-transmission
- **Fast Reconnect**: Automatically restores live or on-demand player to server and server to server connection if disconnected during a broadcast, thus ensuring uninterrupted viewing experience

Microsoft WMS: Additional Features

- Intelligent Streaming: The Player detects network conditions and sends feedback to the Window Media Player, which then adjusts the properties of the stream to maximize quality. Makes use of multiple bit streams, can also use stream thinning

Real Networks Helix

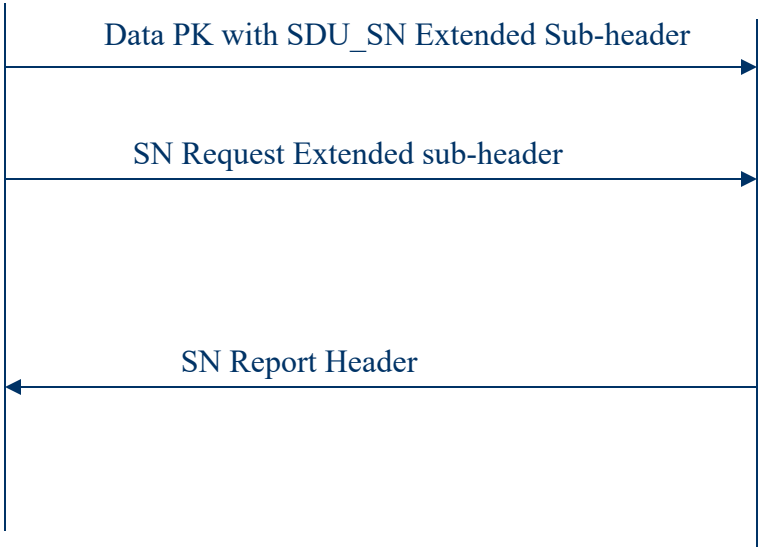
- TrueLive: Reduces client connection and startup time for live broadcasts via RTSP pipelining and improved response times for RTSP messages
- PlayNow: Reduces start up delays for on demand content via system level changes
- Rate Adaptation: Helix Server can adjust data bit rates based on network congestion

WiMAX ARQ

SDSU Sequence Numbering based Packet Recovery

- Main purpose is as a Data Integrity mechanism during a handover (high level description on page 256 of 16e spec)
- Operation:
  - The ASG shall include a SDU SN extended sub-header at least once every 2^p MAC PDUs (p specified in the MAC header and Extended sub-header support TLV)
  - Upon receiving MOB_MSHO_REQ from MS, the serving BS shall include the SDU_SN extended sub-header in MAC PDU at least once before expiration of switching timer (format on page 41)
  - The MS shall maintain MAC SDU sequence number based on information received from the BS. When the MS receives a MAC PDU without the SDU_SN extended sub-header, the MS shall increment the MAC SDU_SN by one for every SDU received. It shall reset the MAC SDU_SN based on the latest value received from the BS

Operation (cont)



- When the MS starts communicating with the target BS after handover, the target BS should assign a UL slot (using UL_MAP IE), for the MS to transmit the LSB of the MAC SDU_SN, using the SN Report MAC header
- The MS shall send two SN Report MAC Headers, that include the MAC SDU_SN for each of the connections that have SN feedback enabled (there can be at most 6 such connections per MS) – format on page 28
- The BS may send the SN Request extended sub-header to explicitly request the MS to send additional SN Report headers – format on page 43

RAN Optimization

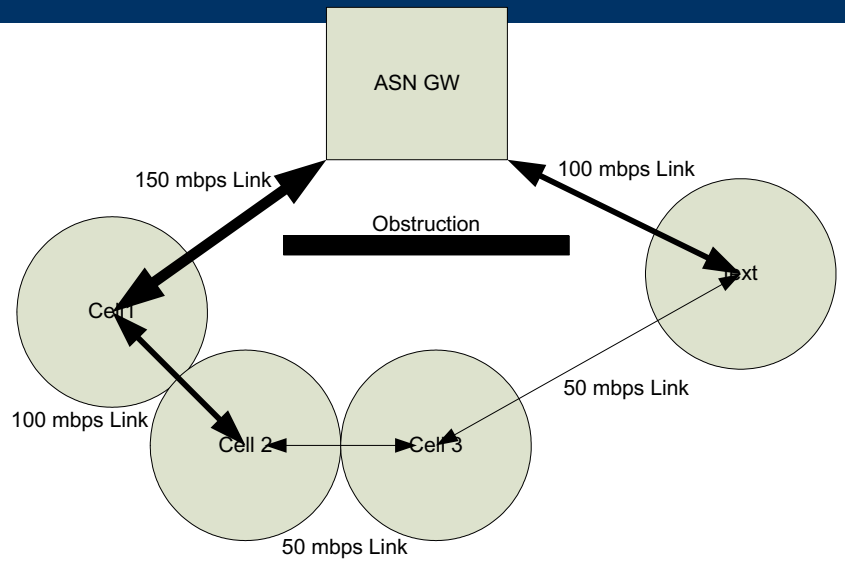
RAN Optimization

- Problem: Backhaul BW is a critical resource that will be in short supply as the airlink bandwidths increase with WiMAX and cell sizes continue to decrease
- State of the Industry: Wireless Carriers spend between 30% to 50% of their OPEX on T1/E1 costs for backhauling their 2G or 3G cell sites. Many Carriers in Europe and Asia use Pt to Pt Wireless links for backhaul
- Apply powerful state of the art Compression Technologies, to dramatically reduce the amount of traffic flowing over the backhaul link.

RAN Optimization

- Requirements (Traditional Compression)
  - The design should support both lossless (GZIP) as well as lossy compression techniques
  - The design should support Datatype specific lossy compression (example: Discarding color information, high frequency components or pixel resolution from an image). Open Source: TranSend (transend.cs.berkeley.edu)
  - The design should adjust the compression ratio as a function of the available bit rate (due to network congestion and/or link variation)and client device type
- Requirements (Network Memory Based Compression aka Byte Compression)
  - The design should support Network Memory based compression, operating at the Transport Layer

RAN Optimization (cont)



Requirements:
- BFE with multiple Ethernet interfaces and multi-hop routing capabilities
- Pt to Pt wireless products with configurable link capacities (eg Dragonwave: Can be configured in increments of 10 mbps, up to 500 mbps)

- Enable a multihop backhaul architecture, with the following benefits:
  - More BW efficient than the Metro Ring architecture, which requires that the BW between nodes be equal to the aggregate of the capacities for all cells
  - It may not be possible to set up pt to pt link between each cell and the ASN GW due to obstructions (pt to pt links require line of site). Setting up smaller hops between cells is more practical.
  - Allows the system to take advantage of statistical multiplexing over the aggregated links
  - Enables redundant backhaul links

Fast Handoffs

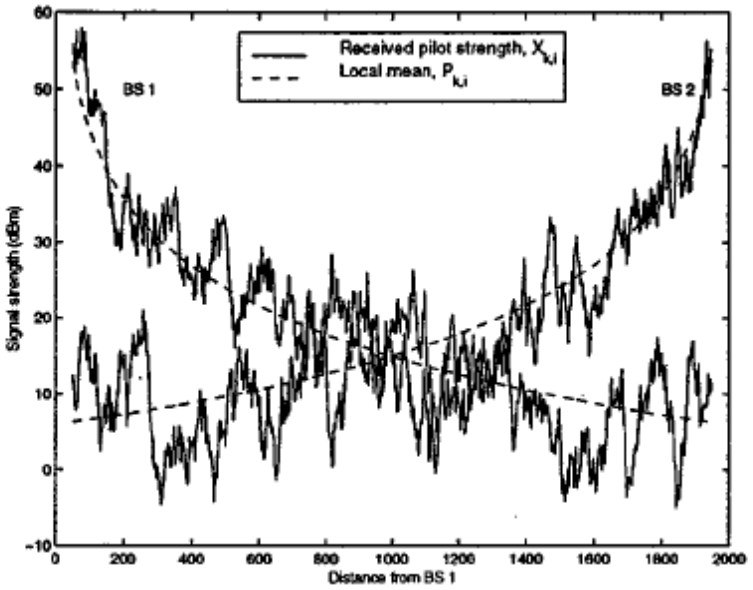
Fast Handoffs

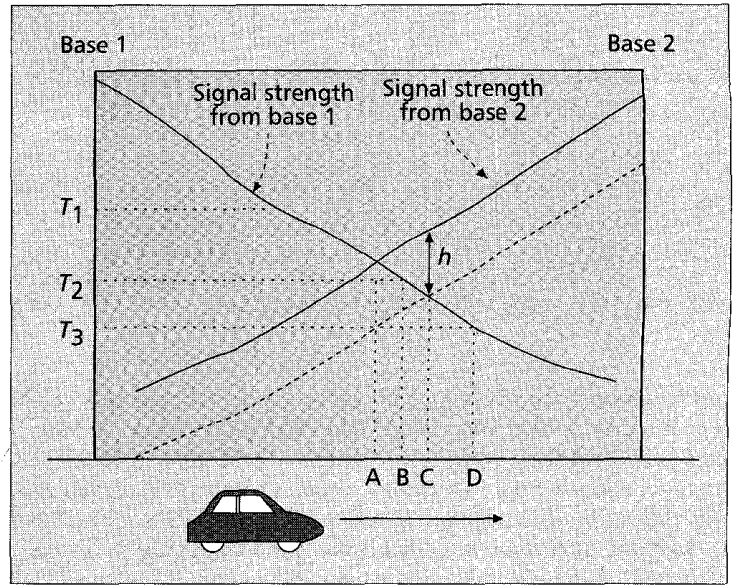
- Problem: The Service Interruption during a Handoff event should not exceed 50 ms, otherwise the connection hiccup can be detected by the user
- State of the Industry: The 50 ms target is achieved in legacy circuit switched networks (2G and 3G), it has yet to be demonstrated for all IP based mobile networks such as WiMAX and WiFi
- Solutions:
  - Optimal Path Prediction Algorithm (PPA): In order to meet the 50 ms handoff target, it is necessary to set up new ASN tunnels and transfer MS context information before the MS attaches to the target BS. This requires that the system estimate the likely targets in advance of the actual handoff, this is the task of the PPA

Fast Handoffs

- Requirements
  - The PPA should not be based on detailed client movement histories
  - The design should take the cell Ping-Pong effect into account
  - The design should incorporate a scheme whereby handoff preparation in neighboring cells can be done in advance, with the actual reservations done during the actual handoff process
  - Investigate the use of GPS based techniques to predict the next cell and the speed of the mobile
  - Investigate Algorithms to minimize the impact of cell scanning (at mobile) on ongoing traffic

Factors Affecting Handoff Performance





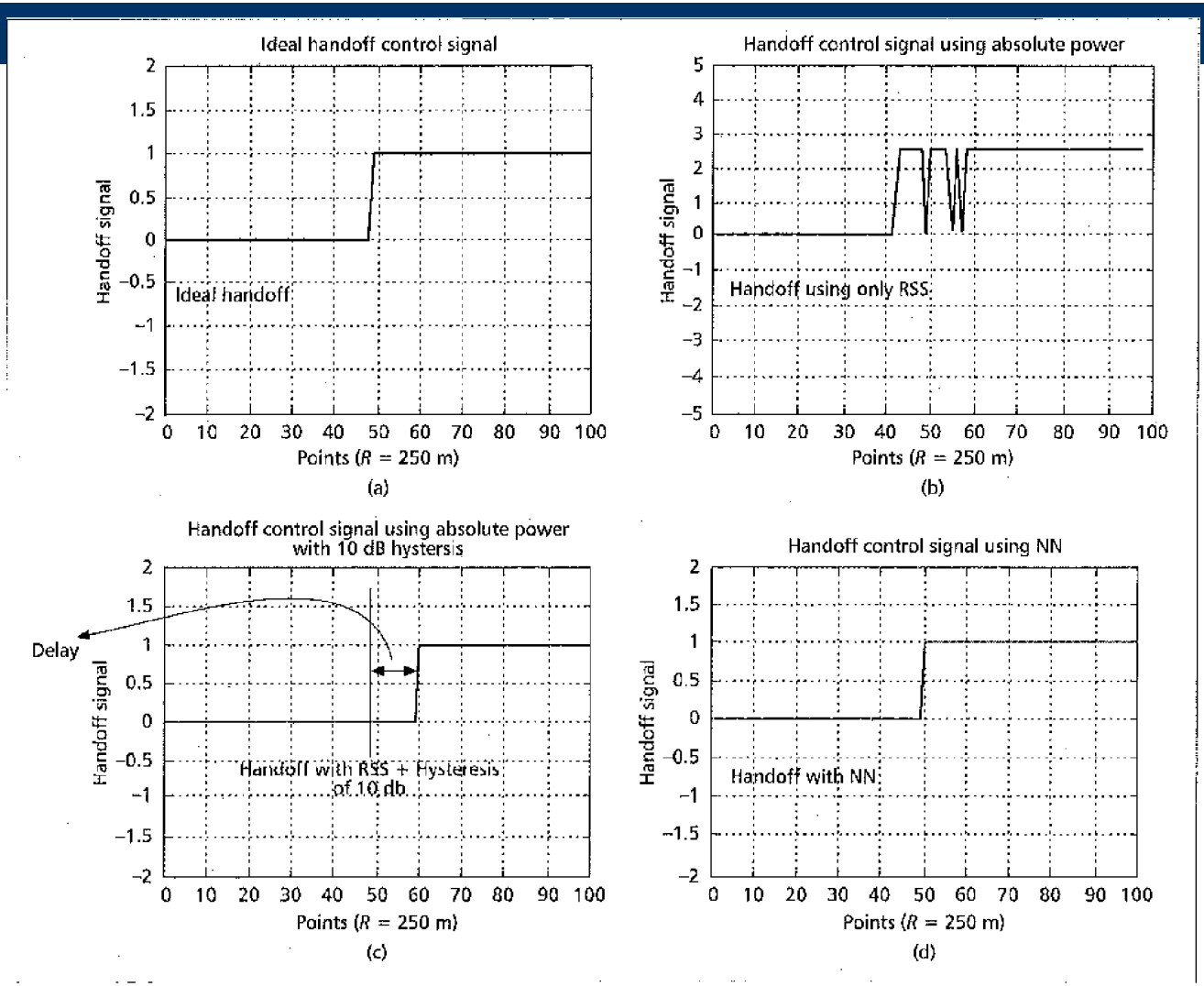
1. Handoff Measurements: RSSI, CINR, ..
2. Handoff Control Point: Network Controlled HO, Mobile Assisted Handoff, Mobile Controlled Handoff. If Network controlled, ASN GW Handoff Control (Profile A) vs BS Handoff Control (Profile C)
3. Handoff Triggers
   - Trigger for Handoff Preparation Phase
   - Trigger for Handoff Action Phase
4. Deciding Target BS Set (Path Prediction)
5. State Information included in the Context Transfer message

Handoff Triggers

- **Handoff Triggers**
  - Relative RSSI,
  - Relative RSSI with Thresholds,
  - Relative RSSI with Thresholds and Hysterisis,
  - Length and Shape of Handoff Window
- Ping Pong Effect: Several Handoffs during the course of traversal between two cells. Can be avoided by setting the Hysterisis and Thresholds appropriately
- Handoff in LoS Conditions: Large Averaging Window and Smaller Hysterisis threshold
- Handoff in NLoS Conditions (can lead to Corner Effect): Smaller Averaging Window and Larger Hysterisis threshold

All These Parameters Remain to be Determined

Effect of Handoff Triggers on Performance: Ping Pong Effect



Handoff Phases

- **Handoff Phases**
  - Handoff Preparation Phase: This is triggered at some threshold in advance of the RSSI threshold value at which connectivity is lost with serving BS. During this phase, MS context information (privacy keys, QoS flow and classification info., TCP state, Compression state etc) are passed to the target BS, and pre-registration done for new GRE tunnels
  - Handoff Action Phase: This is triggered at some RSSI threshold beyond which connectivity is lost with the serving BS. During this phase the MS disconnects from the Serving BS, and ranges with the Target BS, prior to starting data transfer

Handoff Control Point

- Network Controlled Handoff (NCHO)
  - The Network makes the HO decision based on the uplink RSSI measurements from the mobile, at a number of BSs
  - Used in first generation mobile systems such as AMPS
- Mobile Assisted Handoff (MAHO)
  - The MS makes the measurements and the network (BSC or RNC) makes the handoff decisions
  - Used in GSM and 3G
- Mobile Controlled Handoff (MCHO)
  - The MS is completely in control of the handoff process. Used in CDPD, 802.11

Handoffs Using MAHO with Profile A: ASN GW Functions

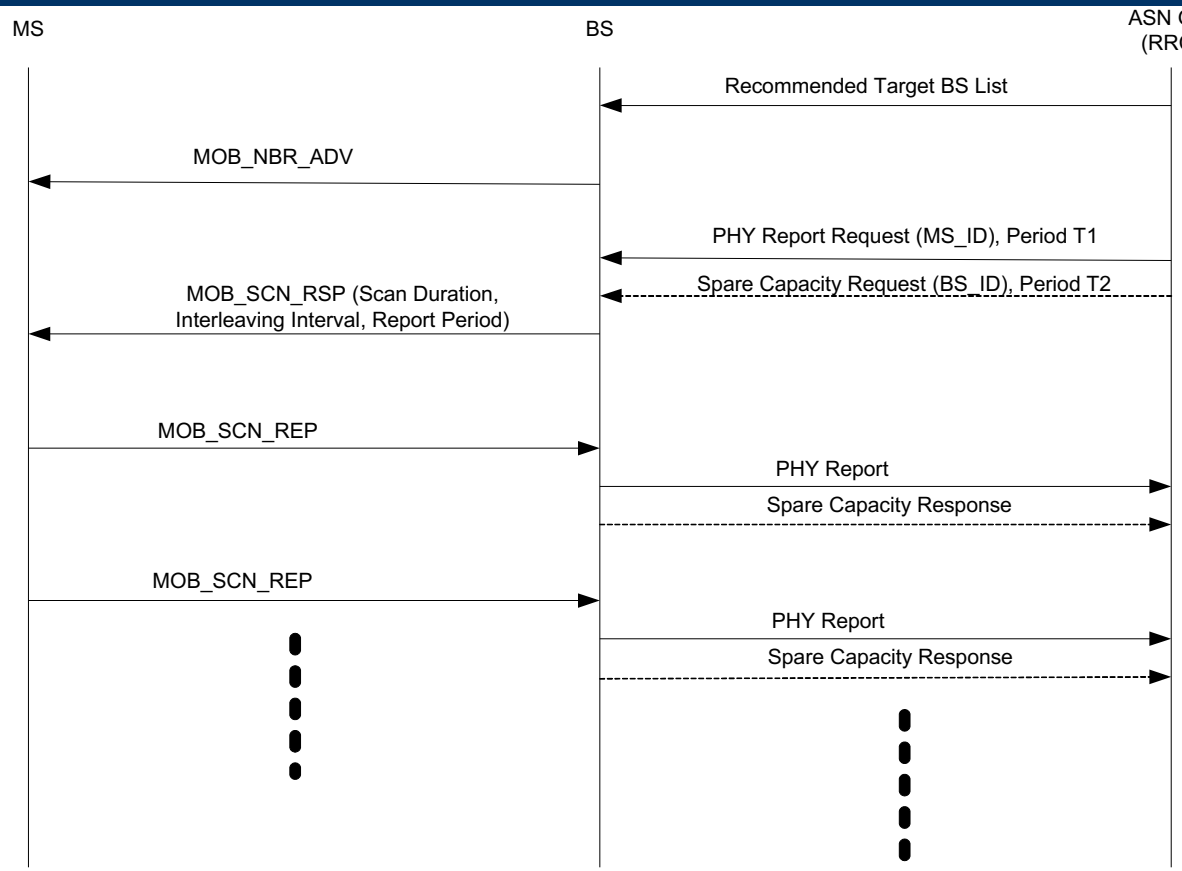
[redacted] Path Prediction Algorithm functions are underlined.

- Collection of RF and Link Utilization information from the MS and BS.
- Selection of (multiple) target BS that should be scanned by the MS.
- Selection of Scanning Frequency (How to minimize scanning overhead)
- Parameter Selection (Trigger Thresholds for HO Preparation and HO Action Phases, Hysterisis threshold)
- Selection of (multiple) target BS set for HO Preparation
- Selection of (single) target BS for HO Action
- Supply MS Context Information to T_BS
- Switch GRE tunnel to T_BS
- Switch MIP tunnel
- Set up QoS connection along new path

Facilitating Information:

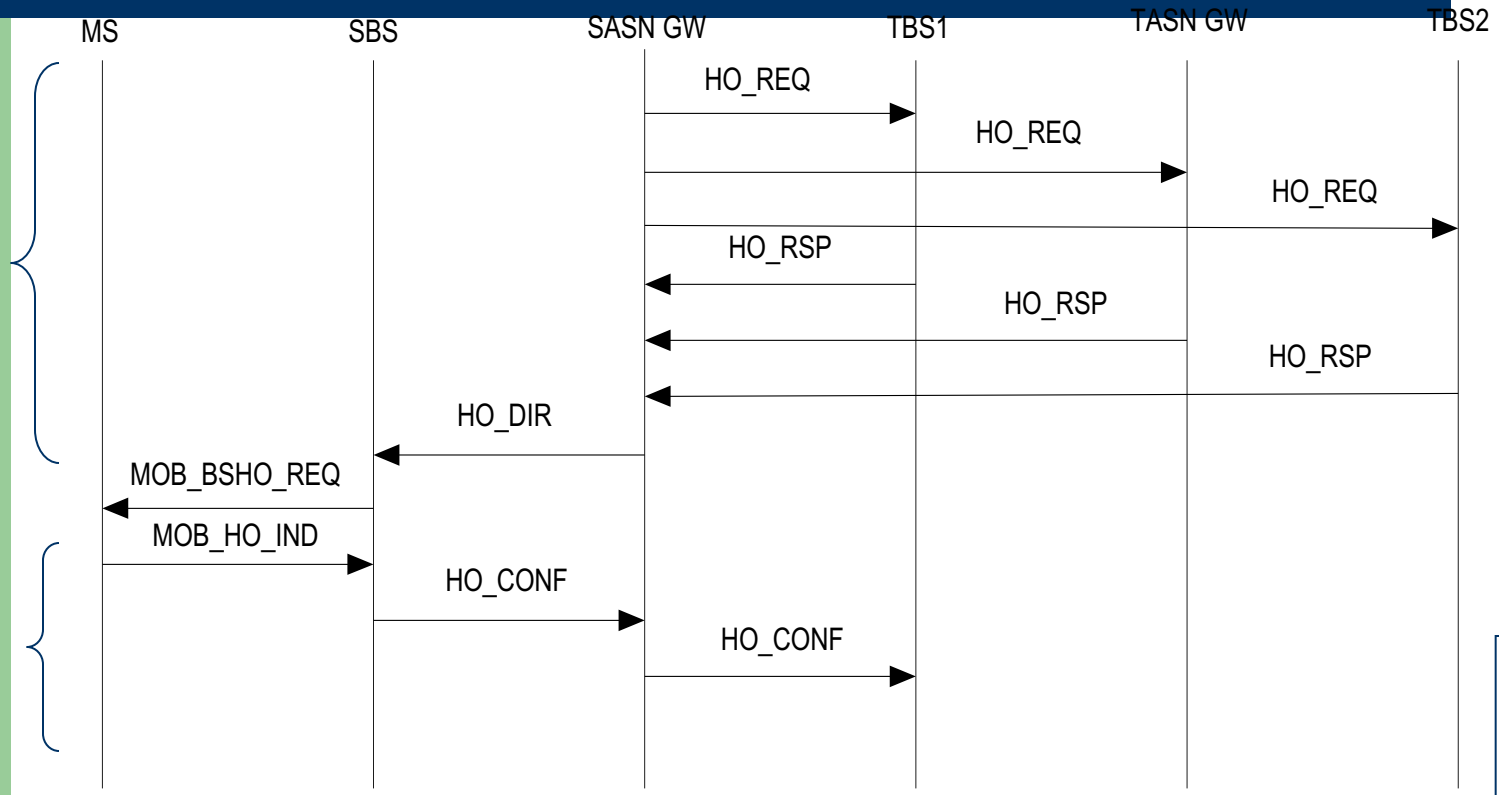
- Network Topology Database
- Link Loading Information (backhaul and airlink, on a per BS basis)
- Maintaining subscriber movement information (on a per BS basis)
- MS Speed Estimation
- GPS Tracking information

MAHO with Profile A: Collecting Measurements



Note: PHY Report Request And PHY Report messages Not defined yet in Stage 3

MAHO with Profile A: Handoff Control Messaging



Note: HO_DIR Message not Yet defined in Stage 3

Other functions that have to be done during HO:

- Context Transfer (Privacy Keys, QoS values per flow, TCP state, Compression State)
- Set up GRE tunnels

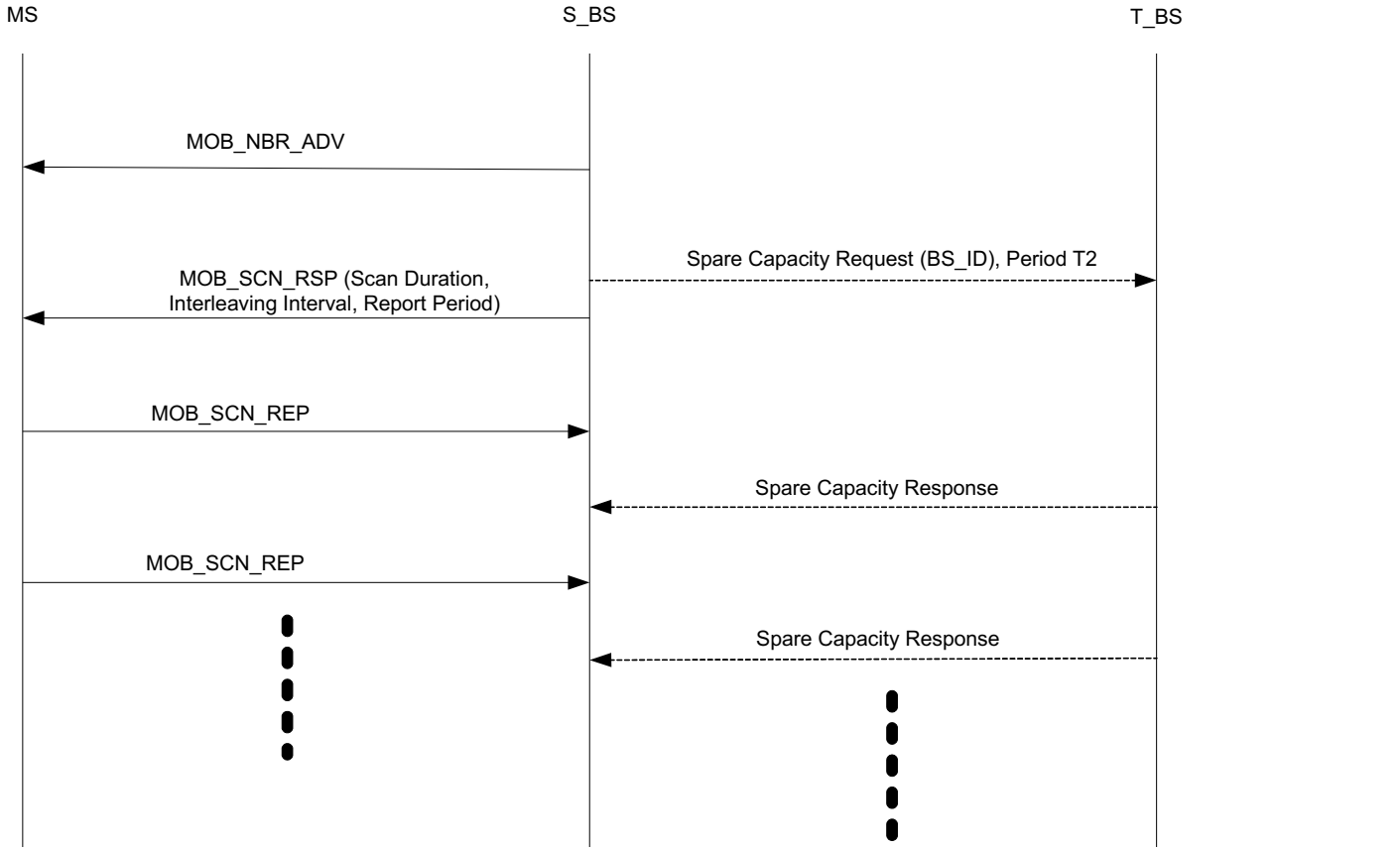
Handoffs Using MAHO with Profile C: ASN GW and BS Functions

- Relay Handoff Control Messages between S_BS and T_BS
- Supply MS Context Information to T_BS and T_ASN GW
- Switch GRE tunnel to T_BS
- Switch MIP tunnel
- Set up QoS connection along new path

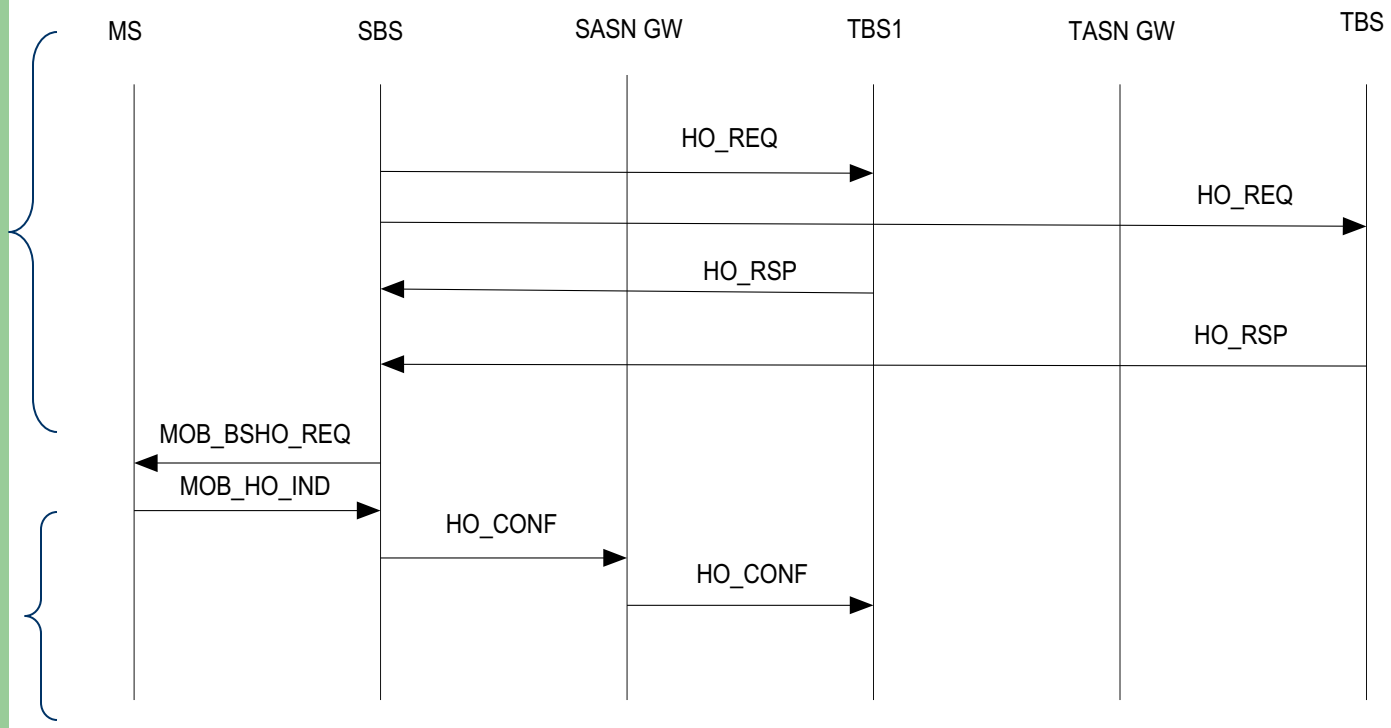
At BS:

- Collection of RF and Link Utilization information from the MS and T_BS.
- Selection of (multiple) target BS that should be scanned by the MS.
- Selection of Scanning Frequency
- Handoff Parameter Selection (Trigger Thresholds for HO Preparation and HO Action Phases, Hysterisis Threshold)
- Selection of (multiple) target BS set for HO Preparation
- Selection of (single) target BS for HO Action

MAHO with Profile C: Collecting Measurements (at S_BS)



MAHO with Profile C: Handoff Control Messaging



Handoffs Using MCHO with Profile C: ASN GW, BS and MS Functions

- Relay Handoff Control Messages between S_BS and T_BS
- Supply MS Context Information to T_BS
- Switch GRE tunnel to T_BS
- Switch MIP tunnel
- Set up QoS connection along new path

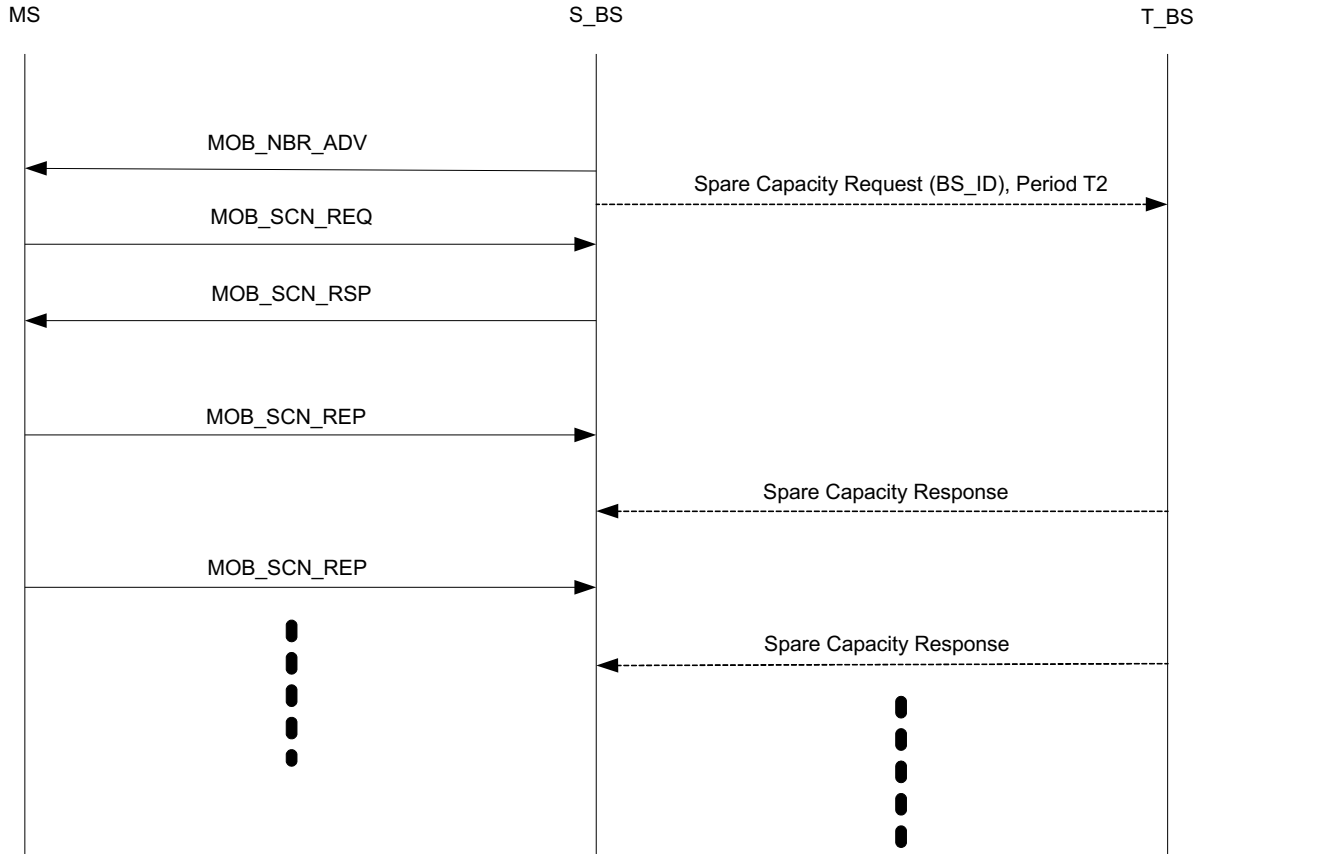
At BS:

- Collection of RF and Link Utilization information from the MS and T_BS.
- Selection of (multiple) target BS that should be scanned by the MS (relayed to MS in NBR_ADV message).
- May modify Scanning Frequency
- May modify target BS set for HO Preparation
- May trigger Handoff Action

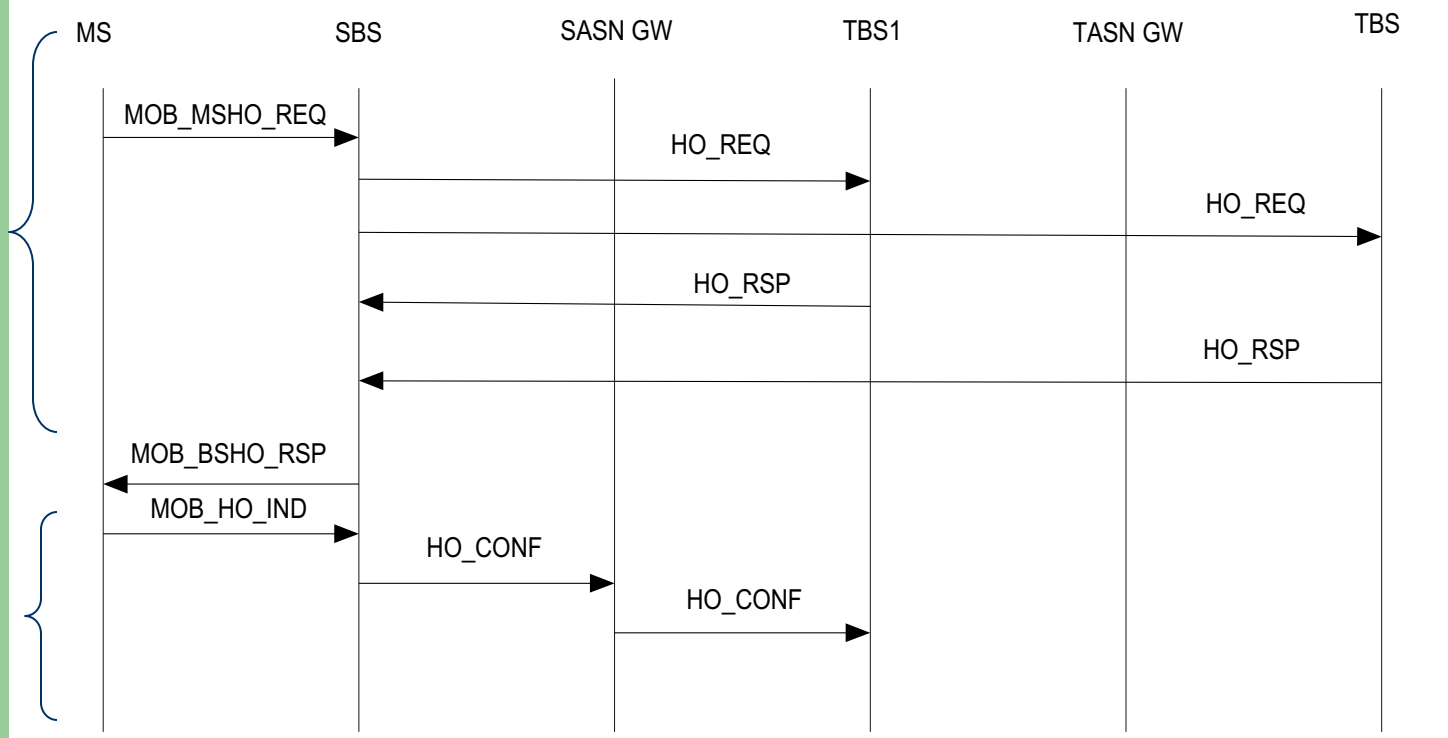
At MS

- Selection of (multiple) target BS to be scanned (with help of NBR_ADV message)
- Selection of Scanning Frequency
- Handoff Parameter Selection (Trigger Thresholds for HO Preparation and HO Action Phases, Hysterisis threshold)
- Selection of (multiple) target BS set for HO Preparation
- Selection of (single) target BS for HO Action

MCHO with Profile C: Collecting Measurements (at S_BS)



MCHO with Profile C: Handoff Control Messaging



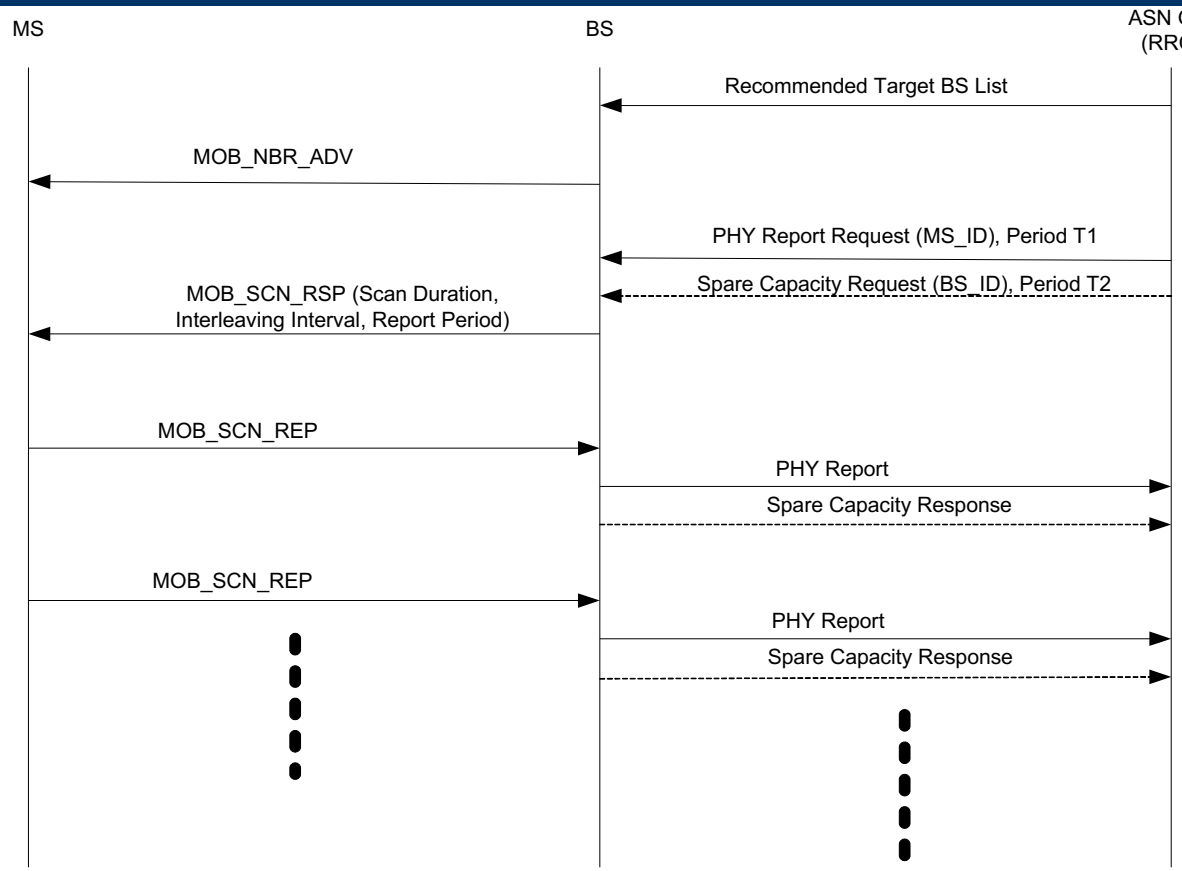
Handoffs Using MCHO with Profile A: ASN GW, BS and MS Functions

- Relay Handoff Control Messages between S_BS and T_BS
- Supply MS Context Information to T_BS
- Switch GRE tunnel to T_BS
- Switch MIP tunnel
- Set up QoS connection along new path
- Collection of RF and Link Utilization information from the MS and T_BS.
- Selection of (multiple) target BS that should be scanned by the MS (relayed to MS in NBR_ADV message).
- May modify Scanning Frequency
- May modify target BS set for HO Preparation
- May trigger Handoff Action

At MS

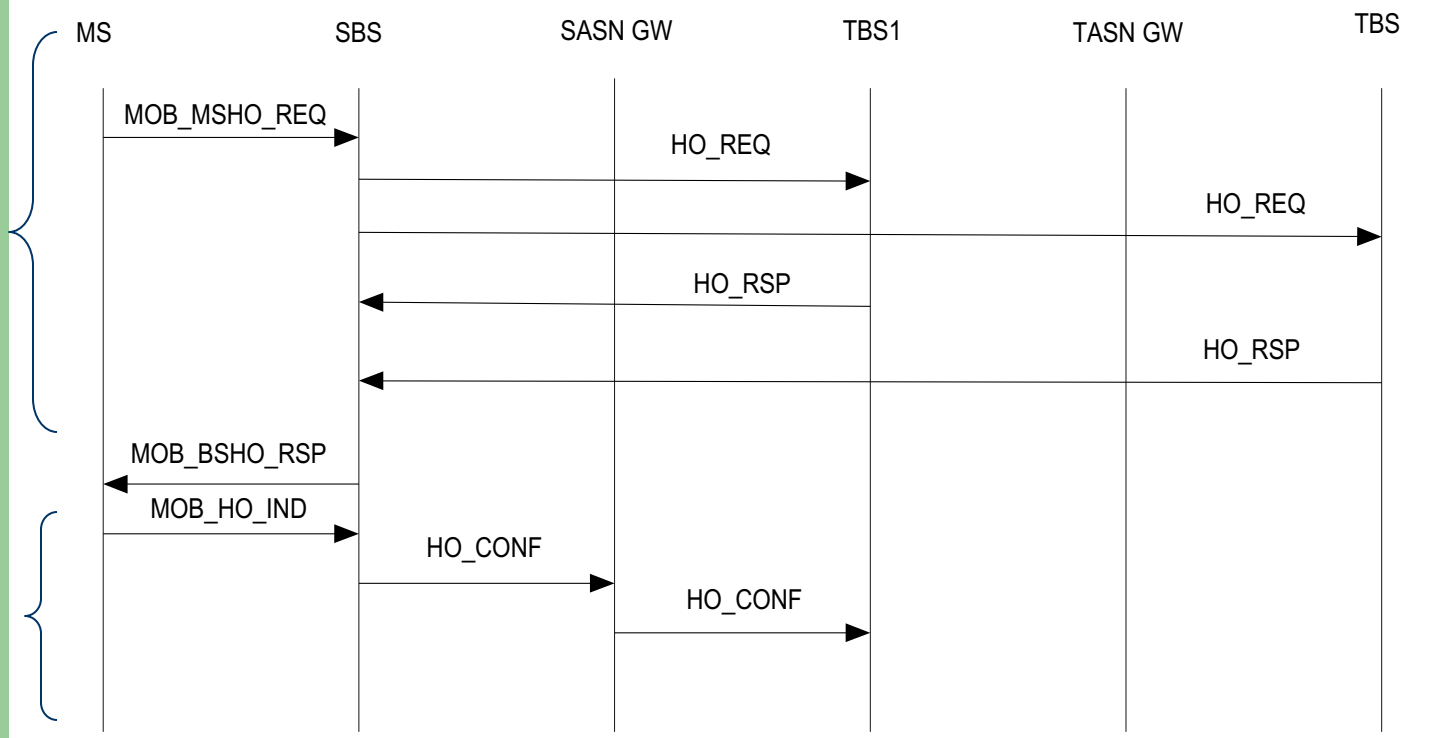
- Selection of (multiple) target BS to be scanned (with help of NBR_ADV message)
- Selection of Scanning Frequency
- Handoff Parameter Selection (Trigger Thresholds for HO Preparation and HO Action Phases, Hysterisis threshold)
- Selection of (multiple) target BS set for HO Preparation
- Selection of (single) target BS for HO Action

MCHO with Profile A: Collecting Measurements



Note: PHY Report Request And PHY Report messages Not defined yet in Stage 3

MCHO with Profile A: Handoff Control Messaging



MCHO vs MAHO

Implications of MCHO:

- The network does not control handoff parameters (thresholds, hysterisis etc) → Poor choice of parameters may lead to ping pong effect, the network design should take these into account
- The network does not control Handoff Preparation trigger point → May lead to the situation that there is not enough time for this
- The network does not control Handoff Action trigger point → May lead to dropped connections

QoS Support

QoS Support

- Problem: QoS reservations are only specified for the airlink (in the WiMAX spec). More ever mobility not only changes the client's point of attachment to the network, but also results in new paths in the backhaul and core network (for the client's traffic), so that QoS reservations for the client along the old path are no longer valid.
- State of the Industry: Existing products provide poor QoS support even for the airlink (with some exceptions, such as Aperto). No vendor provides end to end QoS support, even for the case without handoffs.
- Solutions: Provide end to end BW management and QoS support, using standards based protocols. Also when the client moves around, the reservations are dynamically re-established on the new links.

QoS Support

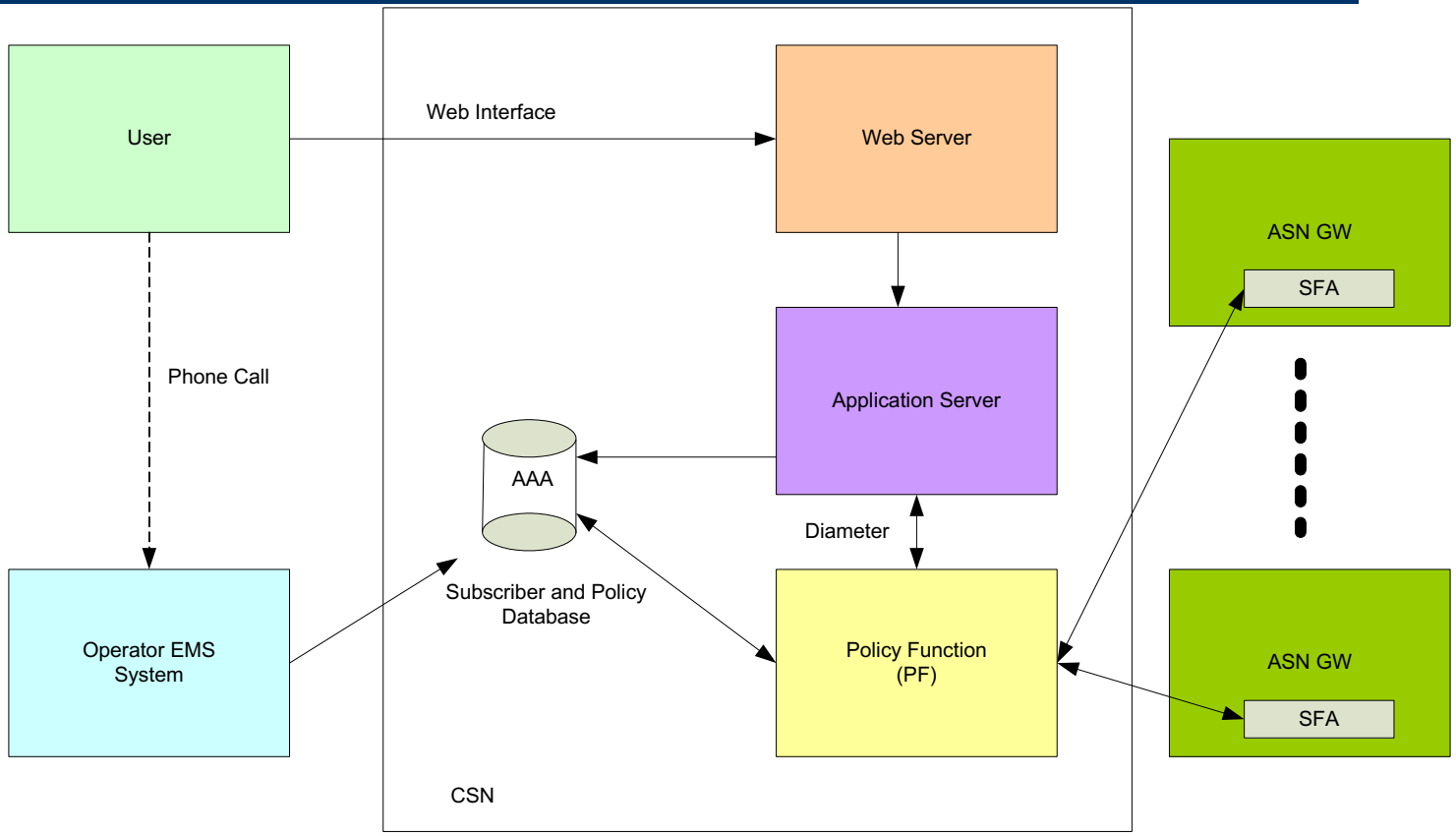
- Requirements
  - The design should integrate Airlink QoS with Backhaul QoS (and at a later stage with Core Network QoS)
  - The design should enable Dynamic QoS flow setup, modification and teardown (to enable features such as BW turbo button and modification to flows due to Link Adaptation and/or Handoffs)
  - The design should automatically transfer QoS flows to new paths established due to handoffs. This should be done in concert with Handoff signaling so that the new QoS flows are established in the Handoff Preparation phase (before data starts to flow)
  - QoS support for Voice should be integrated with SIP/TISPAN signaling, so that the operator can appropriately bill subscribers for voice services
  - Investigate Algorithms to dynamically vary the amount of BW reserved for High Priority flows, as a function of movement patterns of mobiles in neighboring cells

QoS Support: Types of Flows and Scenarios

- Type of Flows
  - Pre-Provisioned Flows: This type of flow is automatically set up when the user enters the network, and stays up for as long as he is active (non idle state). Can be used for general purpose web browsing etc
  - Dynamic Flows: This type of flow is triggered by an Application, such as VoIP or streaming video. It is explicitly set up and torn down by means of signaling messages

- Scenarios
  - Scenario 1: Pre-Provisioned Flow Set Up on Client Network Entry
  - Scenario 2: Flow Modification (Turbo Button)
  - Scenario 3: Voice Call Flow Setup/Teardown
  - Scenario 4: Streaming Video Flow Setup/Teardown
  - Scenario 5: Flow Migration due to Handoff

Scenario 1 and 2



Scenario 1: Initial Network Entry

1. User subscribes to the service using the Service Providers Web Site, or User calls up Service Provider to subscribe for the service
2. In either case the Service Provider creates a profile for the user, containing the flows that the subscriber is entitled to, the bit rates etc. This profile goes into the AAA database
3. When the user powers up his device and does initial registration, the policy information for pre-provisioned flows is downloaded onto the ASN GW at the end of the authentication
4. The SFA module sends the Location Update (LU) message to the PF, and establishes itself as the Anchor SFA for this subscriber. Henceforth all communication from the PF which pertains to this subscriber will go to the Anchor SFA. When the subscriber moves to another cell with a different ASN GW, then the new SFA assumes the role of a serving SFA and registers itself with the Anchor SFA
5. Before applying the policies, the Anchor SFA sends a Policy Decision (PD) message to the PF, to make sure that the policies are still valid
6. In case the policies are valid, the Anchor SFA does the following:
   - Admission Control and Resource Reservation for the pre-provisioned flows on the backhaul link (downlink only)
   - Sends a Resource Reservation (RR) message to the BS. The SFM module on the BS does Admission Control and Resource Reservation for the pre-provisioned flows on the airlink
7. In case both the Admission Control actions are successful, the user can proceed with the rest of the initial entry process (IP Address allocation etc)

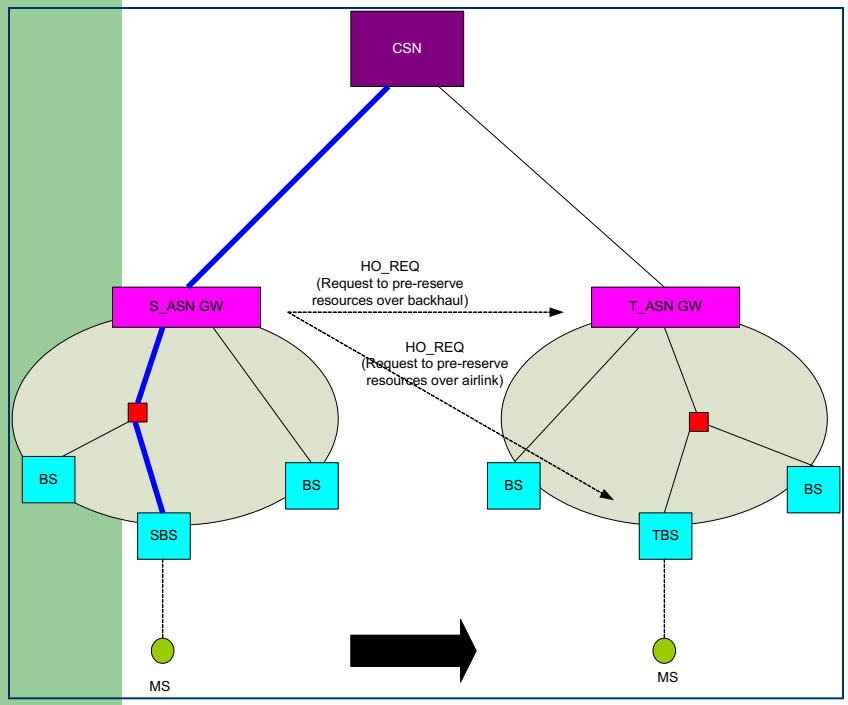
Scenario 2: Turbo Button

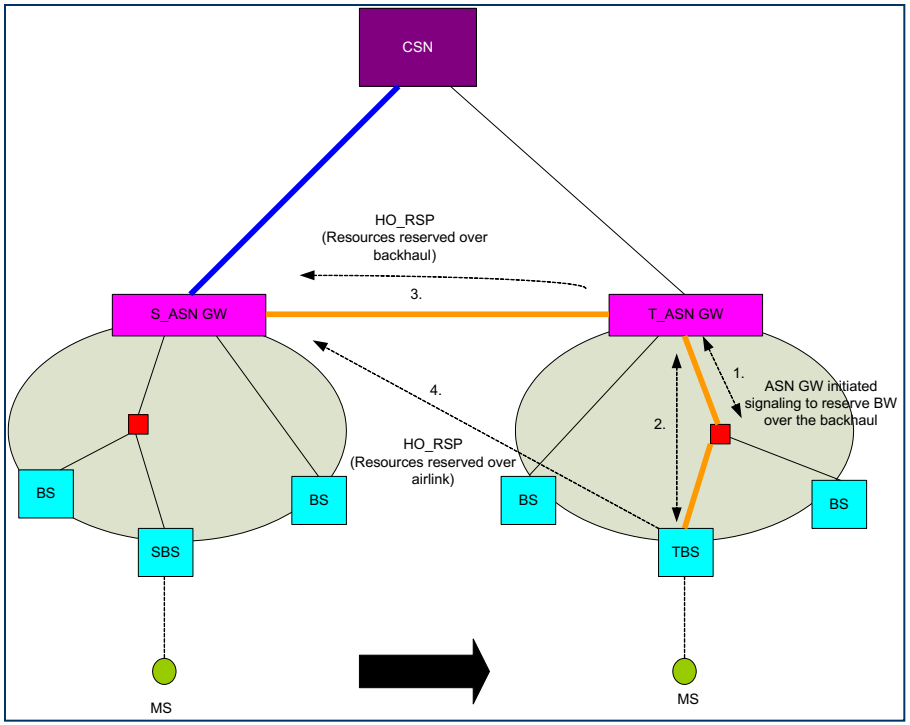
1. User modifies his subscriber profile using the service provider's web site
2. The Application Server sends a message to the PF. If the modifications pertain to pre-provisioned flows, or to dynamic flows that are currently active, the PF sends a RR (modify) message to the Anchor SFA.
3. The Anchor SFA sends the RR message to the Serving SFA
4. The Serving SFA does the following:
   - Admission Control and Resource Reservation for the affected flows on the backhaul link (downlink only)
   - Sends a Resource Reservation (RR) message to the BS. The SFM module on the BS does Admission Control and Resource Reservation for the affected flows on the airlink
5. In case both the Admission Control actions are successful, RR (response) messages are propagated back from the BS → Serving SFA → Anchor SFA → PF

Scenario 3: Voice Calls

1. When the user dials a number, the MS sends a SIP call set up message to the P-CSCF.
2. The P-CSCF sends a message to the PF with information about the call, requesting that the PF co-ordinate resource reservation for the call in the access network
3. The PF consults the Subscriber Database and translates the information into specific service class and bit rates, that it then sends to the Anchor SFA in a RR message
4. In case the MS has moved to a new cell, the Anchor SFA passes on the RR message to its current Serving SFA
5. The Anchor or Serving SFA does the following:
   - Admission Control and Resource Reservation for the voice call on the backhaul link (downlink only)
   - Sends a Resource Reservation (RR) message to the BS. The SFM module on the BS does Admission Control and Resource Reservation for the voice call on the airlink
6. In case both the Admission Control actions are successful, the SFA send a RR (success) message back to the PF, which then signals the P-CSCF that resources have been reserved

Handoffs: Signaling to Pre-Establish QoS over the New Path

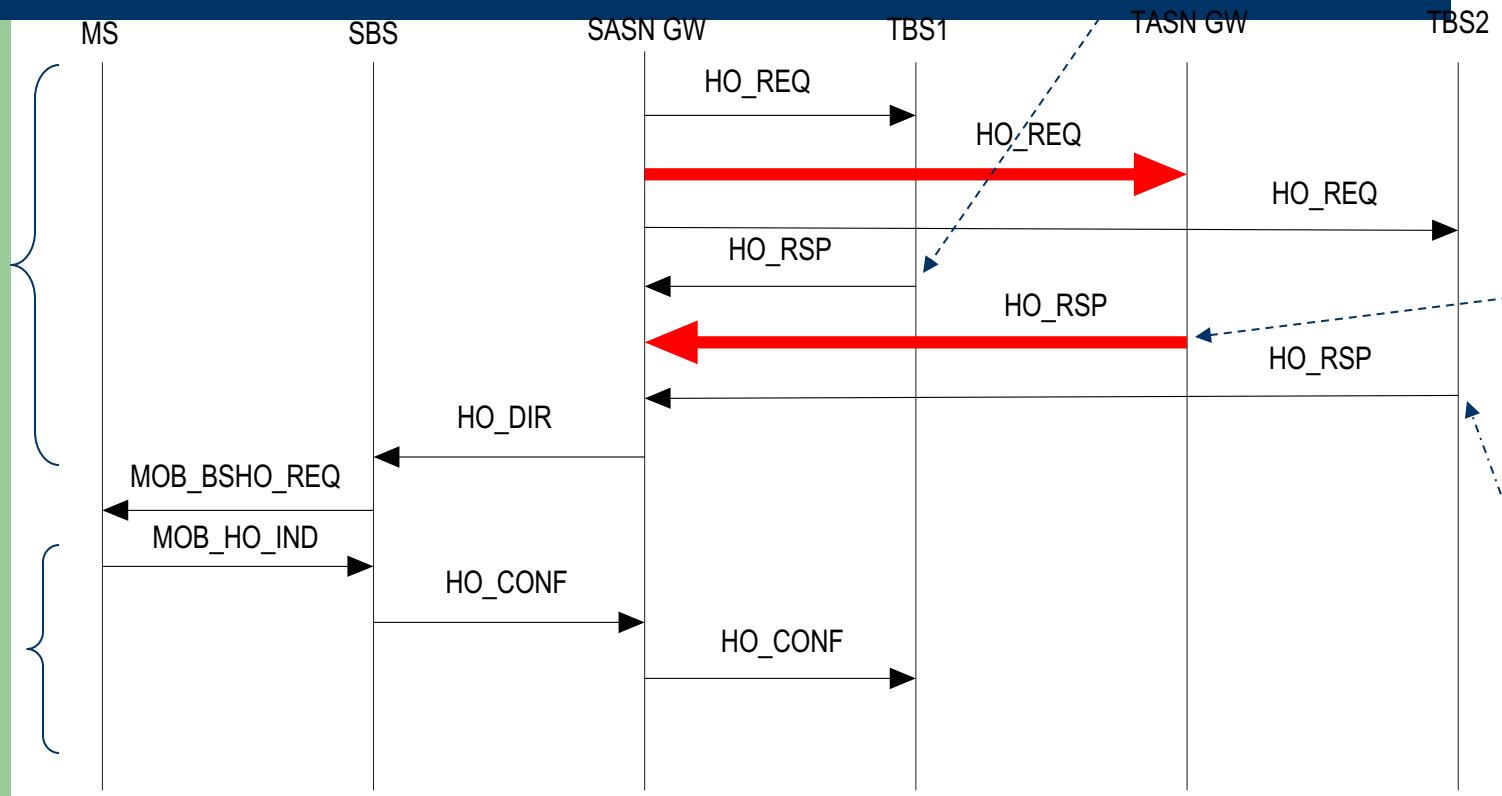




Old Path

New Path

QoS Transfer during Han
MAHO with Profile A

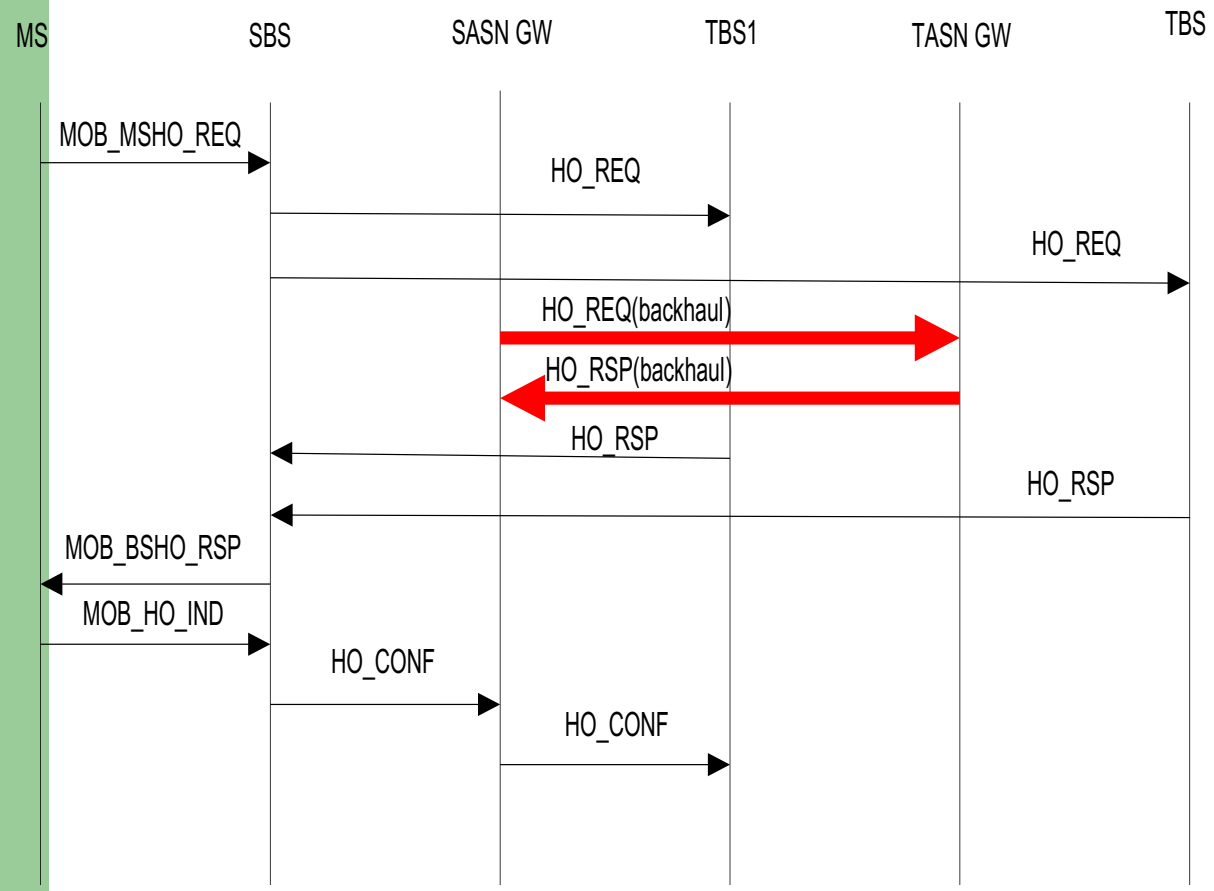


TBS1 reserves BW over airlink1

TASN GW
BW over
between
and TBS

TBS2 reser
BW over ai

QoS Transfer during Handoffs MCHO with Profile C

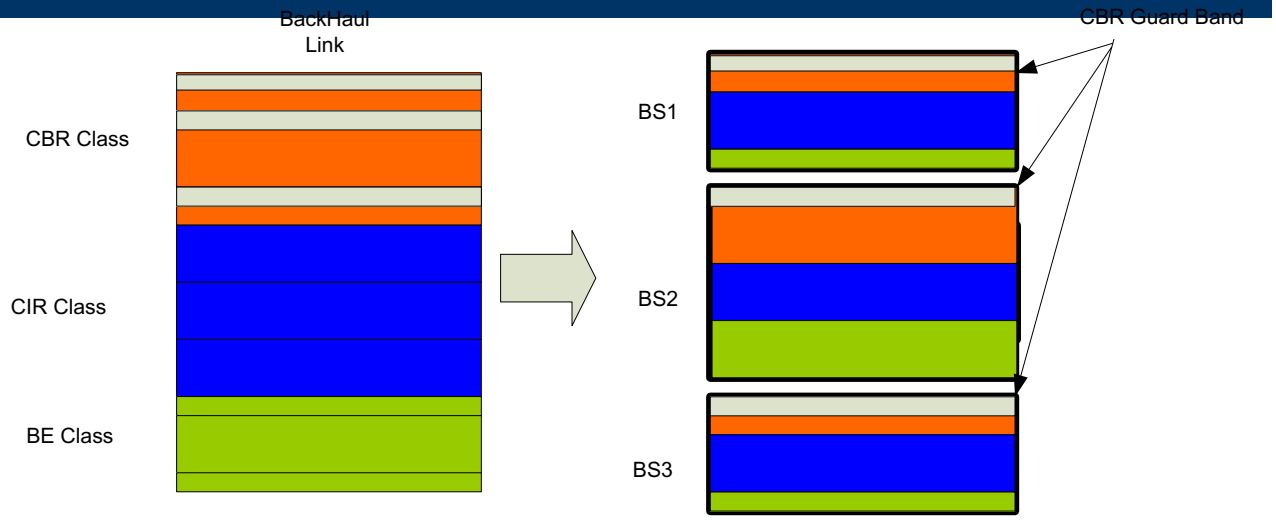


- S_ASN GW snoops On HO_REQ messages And generates its own HO_REQ to reserve Resources over the Backhaul RAN
- If RAN resources are not Available, then the S_ASN GW may modify The contents of the HO_RSP coming from T_BS

Admission Control for CBR Flows

- A CBR flow is admitted if and only if: Sufficient BW is available for that flow Over the Airlink AND Over the Backhaul Link
- Since the backhaul link carries traffic for multiple BSs, should we allow multiplexing of the CBR BW of multiple BSs into a single pool? No, not if there is enough BW in the backhaul for all the BSs, muxing makes sense only if the backhaul capacity is less than the sum of the airlink capacities
- How much CBR capacity should be set aside for the Guard BW? Should the Guard BW for multiple BS be multiplexed?
  - Scheme to dynamically vary the Guard BW as a function of the traffic and mobility patterns in neighboring cells
- The AC module should keep track of the CBR BW usage of each BS. If the BW usage for a BS exceeds the Guard threshold, then the AC module (in the ASN GW) should reject all new CBR flows to that BS. Note that the AC module in the BS (aka SFM) would probably admit those flows, but if the AC module in the ASN GW rejects the flow, then the end to end AC does not succeed

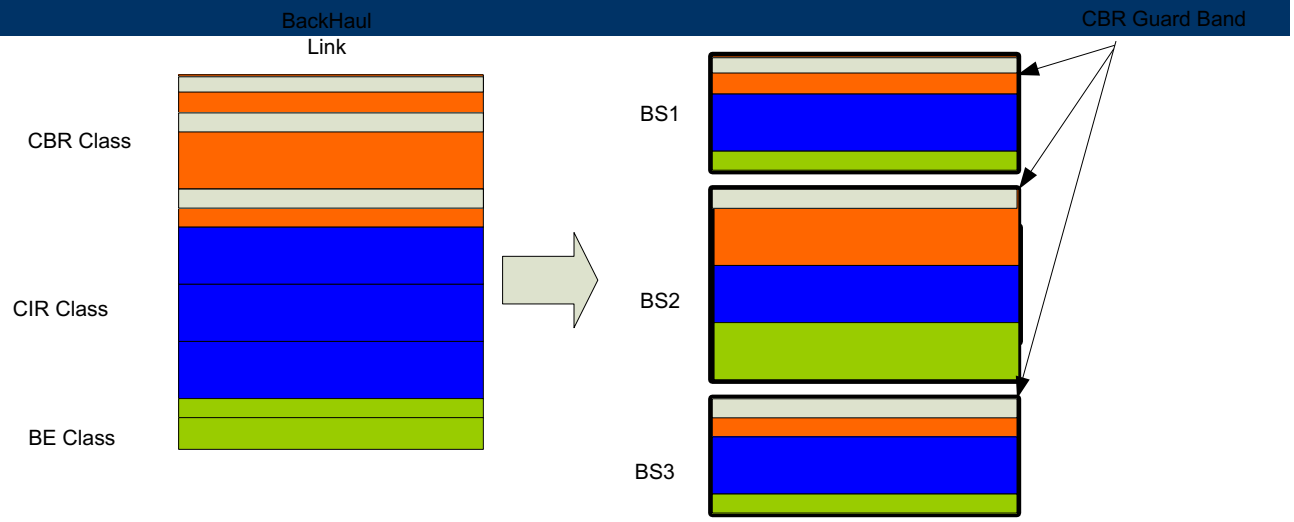
Admission Control at ASN GW: Case 1



Case 1: Backhaul Capacity is equal to or exceeds the sum of the capacities of the BS airlinks

- CBR Admission Control: Done on a per BS basis, takes guard band into account
- CIR Admission Control: Done on a per BS basis
- BE: No Admission Control

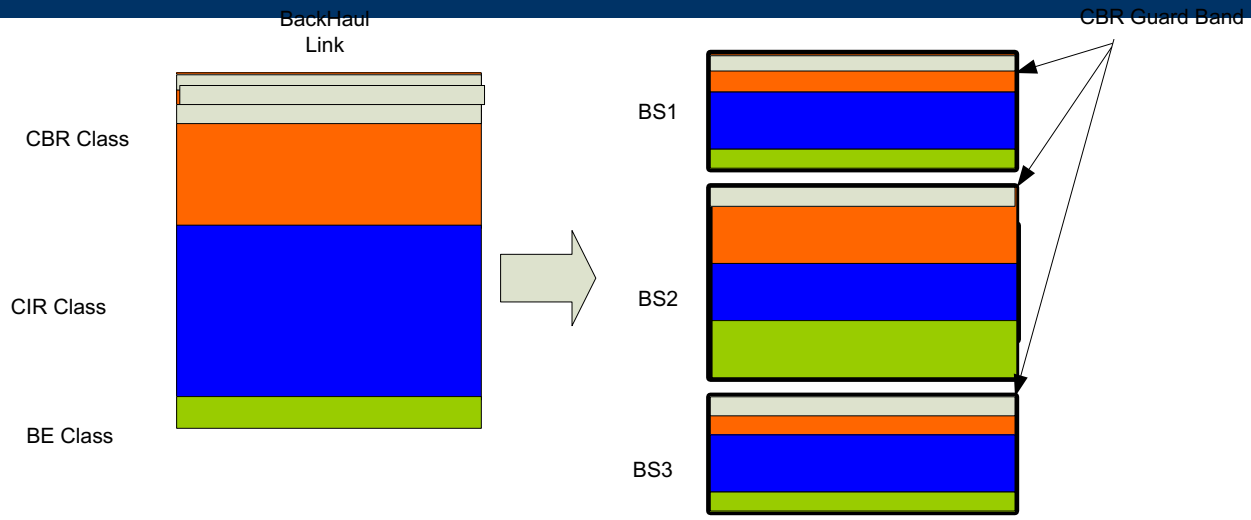
Admission Control at ASN GW: Case 2



Case 2: B < Sum of AirLink Capacities,
B > CBR1+CBR2+CBR3+CIR1+CIR2+CIR3

- CBR Admission Control: Done on a per BS basis, takes guard band into account
- CIR Admission Control: Done on a per BS basis
- BE: No Admission Control. BE capacity on backhaul is reduced

Admission Control at ASN GW: Case 3



Case 3: B < CBR1+CBR2+CBR3+CIR1+CIR2+CIR3

1. Set aside some BW for BE (= C(BE))
2. Reduce the allocation for the CBR and CIR classes by the ratio (B – C(BE))/(CBR1+CBR2+CBR3+CIR1+CIR2+CIR3)

- CBR Admission Control: Done on an aggregate basis, takes guard band into account
- CIR Admission Control: Done on an aggregate basis
- BE: No Admission Control.

Doing AC on an aggregate basis allows the ASN GW to reduce the effects of reduced backhaul capacity

Choosing the Guardband Value

Techniques used in past work on this subject:

1. Derive formulae for the call blocking probabilities, and then choose guardband to ensure that they are below target
   - Assumes Markov models, with Poisson based call generation and handover processes, and exponential channel holding times
   - Does not adapt in response to changing network traffic patterns
2. Use path prediction techniques (GPS based) to estimate number of handoffs that might happen in the near future, and dynamically vary the guardband based on this information

Recommendation:

- Existing techniques are either unrealistic or too complex
- Initial design should use a statically configured guardband
- Investigate adaptive algorithms that vary the guardband based on call blocking information

WiFi WiMAX Integration

WiFi WiMAX Integration

- Problem: WiFi hotspots will continue to exist, as a higher speed complement to Wide Area coverage provided by WiMAX. It should be possible for users to transparently move between these two networks, with smooth handoffs
- State of the Industry: The WiFi industry is independently working on protocols to enable high performance handoffs between WiFi APs. Inter-working protocols between WiFi and WiMAX are yet to be determined
- Solution: We will combine WiFi and WiMAX mobility protocols, to create a three stage network structure: WiMAX handles macro mobility (Mobile IP) and micro mobility (mobility between WiFi controllers), while WiFi handles pico mobility (mobility between WiFi APs).

WiFi WiMAX Integration

- Requirements
  - The design should lead to seamless and low latency handoffs between WiFi and WiMAX networks
  - The design should be such that all the benefits and features in the WiMAX ASN network can be exploited by the WiFi access network
  - The design should support the following configurations (in decreasing order of importance):
    1. WiFi controller connected to the ASN GW over R6
    2. WiFi AP connected to the ASN GW over CAPWAP
    3. WiFi controller backhauled by a WiMAX BS over R1
    4. WiFi AP backhauled by a WiMAX BS over R1

Video Support

- Problem: High Quality Streaming Video over a fully mobile IP based network is an open problem
- State of the Industry: No products available with this capability, at present
- Solution: This capability will be enabled by the following features in our products:
  - Proxy Caching in the ASN GW
  - Optimized Handoffs with PPA
  - Robust Link Control during Handoffs to minimize packet losses
  - Dynamic end to end BW reservations, which change in response to client mobility and wireless link variability

Voice Support

- Problem: IMS and TISPAN compliant voice and other interactive services (Push to Talk, Turbo Button etc)
- State of the Industry: Voice in 2G and 3G networks is carried over the circuit switched infrastructure.
- Solution: This capability will be enabled by the following features in our products:
  - Dynamic end to end BW reservations, triggered by Application Signaling (through SPDF)
  - Optimized Handoffs with PPA
  - Robust Link Control during Handoffs to minimize packet losses