Multihoming: Scheduling, Modelling, and Congestion Window Management

Abdallah Shami
T. Daniel Wallace

Simon Fraser University
July 2013
Talk Outline

• Motivation/Introduction
• A Review of Multihoming Issues using SCTP*
• CMT: Scheduling
• CMT: Modelling
• CMT: Congestion Window Management
• Conclusion

Multihoming

• Computing devices with multiple network interfaces.
  – e.g., the BlackBerry and iPhone include 802.11 (WLAN) and GSM/UMTS (cellular network) technologies.
  – All laptops have WiFi and Ethernet.
Concurrent Multipath Transfer (CMT)

• Goal: Take advantage of multiple network paths, between end-points, to increase application throughput.

• Architecturally: works from the transport layer in the OSI model.

• Congestion control is managed on a per destination basis, but flow control is handled at the session layer (i.e., only one receive buffer).
Stream Control Transmission Protocol

- An Internet Engineering Task Force (IETF) project since 2000 (RFC 4960).
- Implemented in many operating systems (e.g., Linux, Mac OS, FreeBSD, Solaris, Windows).
- TCP Similarities:
  - Reliable transport protocol.
  - Ordered delivery.
  - Implements congestion control and flow control.
- Main difference: SCTP supports multihoming.
  - Allows application data to be transmitted to multiple IP addresses.
  - Most useful for vertical handoffs and network faults.
  - Provides the basics to implement CMT.
System Description

Transport Layer Architecture

Multihomed Network Topology
General Terms & Concepts

• Congestion Window (CWND)
  – A variable that controls the amount of data that can be in flight.

• Send Buffer (SBUF)
  – The amount of memory allocated to accept data from the application layer before it’s send into the network.

• Receive Buffer (RBUF)
  – The amount of memory allocated to accept data from the network before passing it to the application layer.

• Receive Window (RWND)
  – The available space in the RBUF.

• Throughput
  – The rate that data arrives at the receiver.
Multihoming: Problems, Issues, and Challenges

• Handover Management
  – Preemptive Path Selection
  – Fault Tolerant Path Selection
  – Post Handover Synchronization

• Concurrent Multipath Transfer

• Cross Layer Activities
  – Bandwidth estimation
  – Wireless error notifications
  – Network intelligence
Talk Outline

• Motivation/Introduction
• A Review of Multihoming Issues using SCTP*
• CMT: Scheduling
• CMT: Modelling
• CMT: Congestion Window Management
• Conclusion
CMT: Scheduling

- Scheduling & Transmission Basics
- Current Scheduling Approaches for CMT
- On-demand Scheduling
- Performance Results
- Summary & Contributions
Scheduling & Transmission Basics

• Data arrives from the application, fragmented into packets, then waits in the SBUF for a transmission opportunity.

• Transmission opportunities occur when:
  – 1) CWND must be greater than the number of packets it has in flight.
  – 2) RWND is greater than zero.

• Cumulative packet is transmitted.
  – The packet with lowest sequence number in the SBUF that has yet to be sent.
Current Scheduling for CMT

- Naïve Round Robin Scheduling*
  - No intelligence is used during the scheduling process.
  - When only one destination has a transmission opportunity, uses the basic scheduling and transmission technique.
  - When multiple destinations have transmission opportunities, packets are transmitted to each destination in a round robin fashion.

Current Scheduling for CMT

• Naïve Round Robin Scheduling
  – Assume bandwidth is the same on either path, but the delay on the red path is twice as long as the blue path.
  – Both paths always have transmission opportunities.

![Diagram showing CMT scheduling](image)

**Application Layer**

**Transport Layer**

**Network Layer**

**Sender**

**Receive Buffer Blocking**

**Receiver**

**Application Layer**

**Transport Layer**

**Network Layer**
Current Scheduling for CMT

• Bandwidth Aware Scheduler (BAS)*
  – Attempts ordered packet delivery using bandwidth estimates.
  – Scheduling decisions are made before transmission opportunities.
  – When packets arrive at the SBUF, they are assigned to a destination’s send queue.
  – Assignments are based on a reception index.

\[ R(d) = \frac{L(p) + O(d) + S(d)}{B(d)} \]

- \( p \equiv \text{a new packet} \)
- \( d \equiv \text{a destination address} \)
- \( L(p) \equiv \text{returns the size of packet } p \)
- \( O(d) \equiv \text{returns the number of packets (or bytes) in flight to destination } d \)
- \( S(d) \equiv \text{returns the number of packets in the send queue for destination } d \)
- \( B(d) \equiv \text{returns a bandwidth estimate for destination } d \)

Current Scheduling for CMT

- Bandwidth Aware Scheduler
  - Assume bandwidth on the red path is twice the speed of the blue’s, but both experience the same propagation delay.
  - Initially, CWND on the red path is only 1 packet.
On-demand Scheduler

• Goal: Find the cumulative packet (in SBUF) that cannot be delivered to any other destination sooner.

– ODS waits for a transmission opportunity before it makes its scheduling decision.

– Uses bandwidth and propagation delay to manufacture a packet’s estimated time of acknowledgement (ETA).

– Recursively simulates the transmission and acknowledgement of packets in the SBUF.
On-demand Scheduler

Search Algorithm

Copy send buffer.

(1) Search Initialization:
1: let $d^{NS}$ be the destination looking for a packet at the start of a new search;
2: let $t^{NS}$ be the starting time of a new search;
3: let $Q$ be a copy of the send buffer at the start of a new search;
4: let $D$ be a copy of all destination state at the start of a new search;
5: let $D'$ be a set of destinations with earlier ETAs at the start of a new search;
6: goto (2);

Find outstanding packet with lowest ETA and simulate ACK.

(2) Simulating Acknowledgements:
1: if all packets in $Q$ have been transmitted or acknowledged then
2: exit search with failure;
3: else
4: let $a$ be the packet in $Q$ with the earliest ETA such that $a.sent = \text{TRUE}$ and $a.acked = \text{FALSE}$;
5: set $t = a.eta$;
6: if $t < t^{NS}$ then
7: set $t = t^{NS}$;
8: end if
9: set $d = a.dest$;
10: increment $d.pba$;
11: update $d.cwnd$;
12: remove all cumulative packets from $Q$;
13: goto (3);
14: end if

Start a new search.

(3) Calculating ETAs:
1: let $p$ be the packet in $Q$ with the lowest sequence number such that $p.sent = \text{FALSE}$;
2: compute $A(p, d^{NS}, t^{NS})$;
3: for each destination $d$ in $D'$ do
4: compute $A(p, d, t)$;
5: end for
6: if $d^{NS}$ has the lowest ETA at time $t$ then
7: return with $s$
8: else
9: remove any destinations from $D'$ with later ETAs than $d^{NS}$;
10: goto (4);
11: end if

Simulate transmission.

(4) Simulating Transmissions:
1: let $D^{Tx}$ be the set of destinations at time $t$ with a transmission opportunity;
2: if $D^{Tx} \neq \{0\}$ then
3: if $d_{min} \neq d^{Tx}$ then
4: start a new search for $d^{min}$ using $Q$, $D$, and $D'$ at time $t$;
5: goto (1);
6: end if
7: set $p.dest = d^{Tx}$;
8: set $p.eta = A(p, d^{Tx}, t)$;
9: set $p.sent = \text{TRUE}$;
10: end if
11: goto (2);
Performance Results

• Network Topology

• Simulation
  – Implemented each scheduling algorithm in ns-2.
  – Created a variety of network scenarios to evaluate CMT: delay-based disparity, bandwidth-based disparity, loss-based disparity, different RBUF sizes.
  – Simulated a 1 GB file transfer.
Performance Results

For different buffer sizes (RBUF) and packet loss probabilities (p1, p2), the throughput (Mbps) is plotted against the delay on path 1 (ms) for different scheduling and CWND management algorithms. The results show how these parameters affect network performance.

- Naive
- BAG
- ODS
- Path 1
- Path 2
Summary

• Developed a new scheduling algorithm for CMT called On-demand Scheduling (ODS).

• Compared each scheduling algorithm.
  – ODS will often improve performance, especially when the system is constrained by a limited RBUF.
  – BAS is only suitable when the RBUF is very large and preferable when there is a minor disparity in path delays.
  – Naïve scheduling is best when there is minimal difference in delays.

• Evaluated ODS under different network scenarios:
  – delay-based disparity (significant improvement)*
  – bandwidth-based disparity (some improvement)
  – loss-based disparity (still an open problem)

*will revisit later
CMT: Modelling

• Modelling Framework
• Markov Model
• Renewal Model
• Performance Results
• Summary & Contributions
Modelling Framework

• Goal:
  – Given a multihomed system, approximate the throughput of a long-term session employing CMT.

• Parameters:
  – bandwidth (per path)
  – delay (per path)
  – probability of packet loss (per path)
  – RBUF size

• Approach:
  – Model independent SCTP sessions using techniques from TCP literature, then aggregate throughput predictions.

• Assumptions:
  – Perfect scheduling.
  – The CWND must be less than or equal to the RBUF.
Markov Model

• Discrete-time Markov Chain (DTMC)
  – SCTP is represented as a set of discrete transmission rounds.
  – Each round is a state in the Markov chain.
  – During each round some number of packets are transmitted, where some or all of those packets might be lost.

• States:
  – \((\omega, \xi, \tau)\)
  – \(\omega\) = size of the CWND during a round
  – \(\xi\) = number of packets transmitted during a round
  – \(\tau\) = slow-start threshold during a round

• Operating Modes:
  – Congestion Avoidance (CA)
  – Exponential Backoff (EB)
  – Slow-start (SS)
Markov Model

- **States**
  \[ \mathcal{E} = \left\{ (\omega, \xi, \tau) : 2 \leq \omega \leq \omega_{\text{max}}, 1 \leq \xi \leq \omega, \omega \in \mathbb{Z}, \xi \in \mathbb{Z} \right\} \]
  \[ \mathcal{S} = \left\{ (\omega, \xi, \tau) : \omega = \xi = 2^i, 1 \leq i \leq \log_2 \tau, \tau \in \mathcal{I}, \tau > 1, i \in \mathbb{Z} \right\} \]
  \[ \mathcal{E} = \left\{ (0, \xi, \tau) : 2 \leq \xi \leq M, \tau = 1 \right\} \cup \left( \xi = 1, \tau \in \mathcal{I} \right) \}

- **Steady-state probability**
  \[ \pi = \pi \cdot Q \]

- **Throughput**
  \[ \eta = \frac{E[\xi] - E[\gamma]}{E[\delta]} \]
  \[ E[\xi] = \sum_{i \in \{\mathcal{E}, \mathcal{S}\}} \pi(i) \xi(i) + \sum_{i \in \mathcal{E}} \pi(i) \]
  \[ E[\gamma] = \sum_{i \in \{\mathcal{E}, \mathcal{S}\}} \pi(i) \sum_{j} \xi(i) j P(j, \xi(j)) + \sum_{i \in \mathcal{E}} \pi(i) P(1,1) \]
  \[ E[\delta] = \sum_{i \in \{\mathcal{E}, \mathcal{S}\}} \sum_{i' \in \{\mathcal{E}, \mathcal{S}\}} \pi(i) \cdot D(i, i') \]
Renewal Model

• Renewal Theory
  – A stochastic process continually restarts at regular intervals.
  – Formulate a closed-form expression to represent an SCTP session.

• Throughput is approximated by an average interval.
  – \( t \) = length of time of the average interval
  – \( S_t \) = number of packets transmitted during the average interval
  – \( L_t \) = number of packets lost during the average interval

• Operating Modes:
  – Congestion Avoidance (CA)
  – Exponential Backoff (EB)
  – Slow-start (SS)
Renewal Model

- Congestion Avoidance (infinite RBUF, no timeouts)

\[
\eta = \frac{E[S] - E[L]}{E[T]}
\]

\[
E[S] = \frac{1-p}{p} + E[W]
\]

\[
E[L] = \frac{E[W]}{2}
\]

\[
E[T] = (E[X]+1) \cdot \text{RTT}
\]

\[
E[W] = \sqrt{\frac{8(1-p)}{3p} + \frac{1}{9} - \frac{1}{3}}
\]

\[
E[X] = \sqrt{\frac{2(1-p)}{3p} + \frac{1}{36} + \frac{5}{6}}
\]

\[
\eta = \frac{2(1-p) + E[W] \cdot p}{2p \cdot (E[X]+1) \cdot \text{RTT}}
\]
Renewal Model

- Exponential Backoff

\[
\eta = \frac{E[S_{CA}^\text{CA}] - E[L_{CA}^\text{CA}]}{E[T_{CA}^\text{CA}] + P_{TO}} + P_{TO} \cdot E[T_{EB}^\text{EB}]
\]

\[
P_{TO} = \frac{1 - (1 - p)^8 - (1 - (1 - p)^4)(1 - p)^W}{1 - (1 - p)^W}
\]

\[
E[T_{EB}^\text{EB}] = \frac{(1 - p)(1 - 2^M p^M)}{1 - 2p} \text{RTO} + \frac{p^M}{1 - p} \text{RTO}_{\text{max}}
\]
Renewal Model

- Slow-start

\[ \eta = \frac{E[S_{CA}] - E[L_{CA}] + P^{TO} \cdot E[S_{SS}]}{E[T_{CA}]} + P^{TO} (E[T_{EB}^1] + E[T_{SS}]) \]

\[ \begin{align*}
E[S_{SS}] &= E[W] - 1 \\
E[T_{SS}] &= \max(1, \log_2 E[W]) \cdot \text{RTT}
\end{align*} \]
Performance Results

REBUF = 88 KB, d = 40 ms, p = 10^{-3}

REBUF = 88 KB, b = 21 Mbps, p = 10^{-3}

REBUF = 88 KB, b = 21 Mbps, d = 40 ms

b = 21 Mbps, d = 40 ms, p = 10^{-3}

State-space grows too large for Markov model.
Summary

• Created a tractable framework to model the throughput of CMT.
• Developed two different models based on well-known techniques:
  – Discrete-time Markov Chain & Renewal Theory
• Compared the performance of both models with simulated results.
  – Markov model is more accurate but suffers from issues of scalability.
  – Markov model uses Gaussian Elimination to solve an unbounded matrix.
  – Renewal theory is more cost effective, but approximations are not always accurate.
  – Renewal theory approximates throughput using a closed-form expression.
CMT: Congestion Window Management

- CWND Update Policy
- CWND Optimization
- Performance Results
- Summary & Contributions
CWND Update Policy

• $policy_1$: SCTP’s current CWND update policy
  – SCTP grows its CWND by 1 every RTT.
  – CWND is unbounded. Even when flow control stops packets from being transmitted, the CWND continues to grow.

• What impact will $policy_1$ have on CMT?
  – Lowers utilization and throughput potential.
  – One destination address can monopolize the RWND.

• Solution
  – Limit the sum of all CWNDs to the size of the RBUF.
  – Limit the size of a path’s CWND to its corresponding bandwidth delay product (BDP).
  – Apply local or “greedy” optimization.
    o Rank destination addresses according to bandwidth potential.
    o Sets precedence to grow CWND’s of higher ranked destinations first.
CWND Update Policy

- Algorithm name: \textit{policy}_2.

1: let \( d \) be the destination updating its CWND;
2: let \( d.BDP \) be the bandwidth delay product of destination \( d \);
3: let \( d.CWND \) be the congestion window of destination \( d \);
4: let \( d.PBA \) be the value of \textit{partial.bytes.acked} for destination \( d \);
5: let \( \text{CWND}_{\text{sum}} \) be the sum of all congestion windows;
6: let \( \text{RWND}_{\text{max}} \) be the maximum size of the receive window;
7: let \( D \) be the set of all destinations with RTTs higher than \( d \) in descending order;
8: if \( d.PBA \geq d.CWND \) then
9: \quad if \( \text{CWND}_{\text{sum}} \leq \text{RWND}_{\text{max}} \) then
10: \quad \quad if \( d.CWND \leq d.BDP \) then
11: \quad \quad \quad increment \( d.CWND \);
12: \quad \quad \quad else if \( d.CWND > d.BDP \) then
13: \quad \quad \quad \quad decrement \( d.CWND \);
14: \quad \quad end if
15: \quad \quad else if \( d.CWND < d.BDP \) and \( D \neq \emptyset \) then
16: \quad \quad \quad for all \( i \) in \( D \) do
17: \quad \quad \quad \quad if \( i.CWND > 1 \) then
18: \quad \quad \quad \quad \quad decrement \( i.CWND \);
19: \quad \quad \quad \quad \quad increment \( d.CWND \);
20: \quad \quad \quad \quad break;
21: \quad \quad \quad end if
22: \quad \quad end for
23: \quad end if
24: end if

Sum of CWNDS is less than the RBUF.

Limit a destination’s CWND to its BDP.

Decrease CWND of lower ranked dest when CWND of higher ranked dest is blocked.
CWND Optimization

- Optimal performance (i.e. maximum throughput) can be linked to the size of each destination’s CWND.

- Two optimization methods:
  - Dynamic Congestion Window Management (i.e., $policy_2$)
  - Static Congestion Window Management (ILP and heuristic)
CWND Optimization

• Static Congestion Window Management
  – Generate a set of CWND limits to maximize throughput during CMT.
  – Uses CMT performance model (i.e., Markov or renewal model).

• Integer Linear Program

\[
\begin{align*}
\max & \sum_{i}^{n} \eta(b_i, d_i, p_i, c_i^{\max}, t_i^{\max}) \\
\text{s.t.} & \sum_{i}^{n} c_i^{\max} \leq r, \\
& 1 \leq c_i^{\max} \leq \lceil b_i \cdot d_i + 1 \rceil, \\
& c_i^{\max} \in \mathbb{Z}, \\
& \forall i \in N.
\end{align*}
\]

- \( n \) \equiv \text{number of paths}
- \( b_i \) \equiv \text{bandwidth of path } i \text{ (packets per second)}
- \( d_i \) \equiv \text{delay on path } i \text{ (seconds)}
- \( c_i^{\max} \equiv \text{CWND limit on path } i \text{ (packets)}
- \( t_i^{\max} \equiv \text{maximum timeout on path } i \text{ (seconds)}
- \( r \equiv \text{size of the receive buffer (packets)} \)
CWND Optimization

• Heuristic
  – ILP can take a long time to converge.
  – Heuristic reduces the number of searches needed to find a solution.
  – Uses a subset of values when searching for the best set of CWND limits.

\[
\begin{aligned}
k x : 1 \leq x \leq \left\lfloor \frac{a}{k} \right\rfloor, & \quad a = \min\left( r, \left\lceil b_i \cdot d_i + 1 \right\rceil \right), k \leq a \\
\end{aligned}
\]

- \( a \equiv \) total number of different CWND limits
- \( k \equiv \) decreases the search space
- \( b_i \equiv \) bandwidth on path \( i \)
- \( d_i \equiv \) delay on path \( i \)
- \( r \equiv \) size of the receive buffer
Performance Results

• CWND Update Policy
  – Revisit delay-based disparity and compare $policy_1$ vs. $policy_2$

• CWND Optimization
  – Dynamic vs. Static CWND Management
  – Heuristic
Performance Results

CMT: Scheduling, Modelling, and CWND Management
Performance Results

$r = 88$ KB, $b_1 = 21$ Mbps, $d_1 = d_2 = 40$ ms, $p_2 = 10^{-4}$

- $b_2 = 5$ Mbps
- $b_2 = 10$ Mbps
- $b_2 = 15$ Mbps
- $b_2 = 20$ Mbps

$b_1 = 21$ Mbps, $b_2 = 21$ Mbps, $d_1 = d_2 = 40$ ms, $p_1 = p_2 = 10^{-4}$

- $r = 64$ KB
- $r = 88$ KB
- $r = 128$ KB
- $r = 256$ KB

$r = 88$ KB, $b_1 = 21$ Mbps, $b_2 = 20$ Mbps $d_1 = d_2 = 40$ ms, $p_2 = 10^{-4}$

- $p_2 = 10^{-4}$
- $p_2 = 10^{-3}$
- $p_2 = 10^{-2}$
- $p_2 = 10^{-1}$

$r = 88$ KB, $b_1 = 21$ Mbps, $b_2 = 20$ Mbps, $d_2 = 60$ ms, $p_2 = 10^{-4}$

- $d_2 = 30$ ms
- $d_2 = 40$ ms
- $d_2 = 50$ ms
- $d_2 = 60$ ms
Performance Results

- **Heuristic**
  - Parameters: $r = 128$ KB, $p_1 = p_2 = 10^{-4}$, $d_1 = d_2 = 40$ ms, $b_2 = 10$ Mbps, $k$ (variable), $b_1$ (variable).

Higher values of $k$ yield higher throughput. Lower values of $k$ take less time to find a solution.
Summary

• Developed a new CWND update policy for CMT.
  – Compared \( policy_1 \) to \( policy_2 \) under delay-based disparity.

• Created an ILP to solve the static CWND management optimization problem.
  – Compared dynamic and static CWND management under different network scenarios.
  – Static CWND management yields better results but requires system knowledge (e.g., loss rate) and increases computational complexity.

• Reduced computational complexity by developing a simple heuristic.
  – Evaluated our heuristic using various subsets of CWND limits.
  – Using larger values of \( k \) lowers performance capabilities but also reduces computational requirements.
Open Challenges
Challenges

• CMT: Scheduling
  – Problem: ODS is a search algorithm that has some computational requirements.
  – Develop a closed-form expression that imitates ODS.
  – Implement ODS in the Linux kernel.
Open Challenges

• **CMT: Modelling**
  
  – Problem: perfect scheduling was assumed to avoid receive buffer blocking due to loss-based disparity.
  
  – Incorporate the effects of loss-based disparity into the model.
Open Challenges

• CMT: CWND Management
  – Problem: short term gains are not considered during the static optimization process.
  – Include the short term gains into the CMT model for static CWND management.
  – Develop a solution method using a metaheuristic (e.g., simulated annealing, tabu search, genetic algorithms).
  – Formulate an optimal decision policy using a Markov Decision Process (MDP).