RC3
Recursively Cautious Congestion Control

Radhika Mittal, Justine Sherry,
Sylvia Ratnasamy, Scott Shenker

UC Berkeley
Roadmap

• Isn’t congestion control a solved problem?

• Scope for performance gains

• Design Details

• Simulation Results

• Linux Implementation and Evaluation

• Challenges and Future
Roadmap

• Isn’t congestion control a solved problem?
  
  • Scope for performance gains
  
  • Design Details
  
  • Simulation Results
  
  • Linux Implementation and Evaluation
  
  • Challenges and Future
Short Flow Completion Time

- “Being fast really matters. Users really respond to speed.”
  - 0.5 sec delay caused a 20% drop in traffic – Google
  - 2 sec slowdown changed queries/user by -1.8% and revenue/user by -4.3% – Bing
  - 5 sec speedup resulted in a 25% increase in page views and 7-12% increase in revenue – Shopzilla

- James Hamilton’s Blog
RC3 in a nutshell

Send *additional* packets from the flow using low priority service (*WQoS*), filling up only the *spare capacity* in the network

- 40-80% Reduction in Flow Completion Time
- No harm to the regular high priority traffic
- Better use of Network Resources
Example Scenario

Sender

Network Provider

Receiver
Network Provider Viewpoint

What if I get a burst of traffic in peak hours or a failure occurs?

Must overprovision 30-50% average link utilization
The network might be very congested!

Must ramp-up cautiously
TCP

- Slow Start (Wasted Capacity)
- Congestion Avoidance (Link Fully Utilized)
- (Long FCT)
The Root Cause

Two Goals of Congestion Control
- Fill the pipe for high throughput
- Do no harm to other flows

Traditional Approach
- Single mechanism tries to balance the two conflicting goals

**RC3:** Decouple these goals using priorities
- Fill the pipe at lower priority
- Do no harm at higher priority
RC3 in action

Additional Packets at Low Priority Fill the Pipe
Regular TCP at High Priority

Flow Completes Sooner
Example: FCT with Slow Start

7 packets flow (with initial congestion window of 1 segment) completes in 3RTTs under slow start
Example: FCT with RC3

Remaining 6 packets sent at lower priority with the 1st packet
Flow completes in 1RTT
Roadmap

• Isn’t congestion control a solved problem?
  – Conflicting goals of high throughput and friendliness decoupled through priorities

• Scope for performance gains

• Design Details

• Simulation Results

• Linux Implementation and Evaluation

• Challenges and Future
Theoretical Model

% FCT Reduction

TCP: 1 RTT
RC3: 1 RTT
TCP: $\log\left(\frac{N}{i}\right)$ RTTs
RC3: 1 RTT
TCP: $\log\left(\frac{A \times RTT}{i}\right)$ RTTs + $\frac{N}{A}$
RC3: 1 RTT + $\frac{N}{A}$

Available Capacity
$A = (1-u)BW$

Initial Congestion Window

Flow Size ($N$)
Parameter Sensitivity: $A \times RTT$

Flow Size $(N)$

- TCP: $1$ RTT
- RC3: $1$ RTT

- TCP: $\log\left(\frac{N}{i}\right)$ RTTs
- RC3: $1$ RTT

- TCP: $\log\left(\frac{A \times RTT}{i}\right)$ RTTs + $\frac{N}{A}$
- RC3: $1$ RTT + $\frac{N}{A}$

Available Capacity $A = (1-u)BW$

Initial Congestion Window

% FCT Reduction

Higher $A \times RTT$
Roadmap

• *Isn’t congestion control a solved problem?*
  – *Conflicting goals of high throughput and friendliness decoupled through priorities*

• *Scope for performance gains*
  – *Increases with increasing RTTxBW*

• **Design Details**

• *Simulation Results*

• *Linux Implementation and Evaluation*

• *Challenges and Future*
WQoS Implementation

Routers offer several layers of worse service
- Use Priority Queues
- Support already present

Packets carry priority (possibly) in DSCP field
- Priority 0 – default (highest)
- Priority 1, Priority 2, Priority 3,...
RC3 Design

RC3 runs two parallel control loops

– TCP control loop
  Transmits packets that obey unmodified TCP logic at highest priority

– Recursive Low Priority (RLP) control loop
  Transmits additional packets at low priority
What packets are sent at low priority?

Minimum overlap between packets sent by the two control loops for maximum gains
- RLP starts from the last packet in buffer
- Goes in reverse order

TCP Control Flow  →  RLP Control Flow

0  1  2  3  4  ....  ....  N-1  N

High Priority  ─── Low Priority
Single Flow

Sender

Network Provider

Receiver

Bottleneck

BWxRTT = 9 packets
Multiple Flows?

Sender 1

Sender 2

Network Provider

Receiver
Router’s Priority Queue
Multiple Flows?

Sender 1

Sender 2

Network Provider

1 1 1 1 1 1 1 0 0

Receiver
Recursively Cautious Congestion Control

- Use multiple priority levels
- Send exponentially larger number of packets at each priority level
RC3 Design: Quick Recap

Two parallel control loops
  – Regular TCP
  – Recursive Low Priority (RLP)

Minimum overlap between the two control loops
  – Send low priority packets from the end in reverse order

Max-min fairness across flows
  – Use multiple priority levels
Roadmap

• Isn’t congestion control a solved problem?
  – Conflicting goals of high throughput and friendliness decoupled through priorities

• Scope for performance gains
  – Increases with increasing RTTxBW

• Design Details
  – Additional packets sent backwards from the end using multiple low priority levels

• Simulation Results

• Linux Implementation and Evaluation

• Challenges and Future
Simulation Setup

- Multi-hop **Internet-2** network topology
  - 10 core nodes, 100 end hosts
- **1Gbps** bottleneck bandwidth
- **40ms** average RTT
- Baseline is **30%** average link utilization
- Pareto flow size distribution with Poisson inter-arrival
- Initial Congestion Window of **4 segments**
Comparing baseline simulation results with the theoretical model

% FCT Reduction

Flow Size ($N$)

$N_i \times \text{RTT}$
Baseline

The diagram shows the reduction in average FCT for different flow sizes, comparing theoretical results with simulation results. The x-axis represents the flow size in KB and MB, while the y-axis represents the percentage reduction in average FCT. The red bars represent the theoretical result, and the gray bars represent the simulation result.
Baseline

Flow Size < 4MSS
- No RC3 packets sent

Benefits because high priority congestion is reduced
Baseline

Model does not account for queuing delays and drops.
Baseline

<table>
<thead>
<tr>
<th>Flow Size</th>
<th>Regular TCP</th>
<th>RC3</th>
<th>% Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5KB</td>
<td>0.135s</td>
<td>0.076s</td>
<td>43.54%</td>
</tr>
<tr>
<td>3KB</td>
<td>0.135s</td>
<td>0.076s</td>
<td>43.54%</td>
</tr>
<tr>
<td>4KB</td>
<td>0.135s</td>
<td>0.076s</td>
<td>43.54%</td>
</tr>
<tr>
<td>7.5KB</td>
<td>0.135s</td>
<td>0.076s</td>
<td>43.54%</td>
</tr>
<tr>
<td>10KB</td>
<td>0.135s</td>
<td>0.076s</td>
<td>43.54%</td>
</tr>
<tr>
<td>60KB</td>
<td>0.443s</td>
<td>0.114s</td>
<td>74.35%</td>
</tr>
<tr>
<td>100KB</td>
<td>0.443s</td>
<td>0.114s</td>
<td>74.35%</td>
</tr>
<tr>
<td>200KB</td>
<td>0.443s</td>
<td>0.114s</td>
<td>74.35%</td>
</tr>
<tr>
<td>400KB</td>
<td>0.443s</td>
<td>0.114s</td>
<td>74.35%</td>
</tr>
<tr>
<td>1.5MB</td>
<td>0.443s</td>
<td>0.114s</td>
<td>74.35%</td>
</tr>
<tr>
<td>3MB</td>
<td>0.443s</td>
<td>0.114s</td>
<td>74.35%</td>
</tr>
</tbody>
</table>

Theoretical Result

Average Over Flows

<table>
<thead>
<tr>
<th>Flow Size</th>
<th>Regular TCP</th>
<th>RC3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5KB</td>
<td>0.135s</td>
<td></td>
</tr>
<tr>
<td>3KB</td>
<td>0.135s</td>
<td></td>
</tr>
<tr>
<td>4KB</td>
<td>0.135s</td>
<td></td>
</tr>
<tr>
<td>7.5KB</td>
<td>0.135s</td>
<td></td>
</tr>
<tr>
<td>10KB</td>
<td>0.135s</td>
<td></td>
</tr>
<tr>
<td>60KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>100KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>200KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>400KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>1.5MB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>3MB</td>
<td>0.443s</td>
<td></td>
</tr>
</tbody>
</table>

Average Over Bytes

<table>
<thead>
<tr>
<th>Flow Size</th>
<th>Regular TCP</th>
<th>RC3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>3KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>4KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>7.5KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>10KB</td>
<td>0.443s</td>
<td></td>
</tr>
<tr>
<td>60KB</td>
<td>0.114s</td>
<td></td>
</tr>
<tr>
<td>100KB</td>
<td>0.114s</td>
<td></td>
</tr>
<tr>
<td>200KB</td>
<td>0.114s</td>
<td></td>
</tr>
<tr>
<td>400KB</td>
<td>0.114s</td>
<td></td>
</tr>
<tr>
<td>1.5MB</td>
<td>0.114s</td>
<td></td>
</tr>
<tr>
<td>3MB</td>
<td>0.114s</td>
<td></td>
</tr>
</tbody>
</table>

% Reduction in avg FCT
Comparing RC3 with other schemes
RC3 in comparison

- Increasing the initial congestion window
- Rate Control Protocol (RCP)
RC3 in comparison

- Increasing the initial congestion window
- Rate Control Protocol (RCP)
Comparison: Increasing InitCWnd

TCP: \(1\) RTT
RC3: \(1\) RTT

TCP: \(\log\left(\frac{N}{i}\right)\) RTTs
RC3: \(1\) RTT

TCP: \(\log\left(\frac{A \times RTT}{i}\right)\) RTTs + \(\frac{N}{A}\)
RC3: \(1\) RTT + \(\frac{N}{A}\)

% FCT Reduction

Flow Size \((N)\)
Comparison: Increasing InitCwnd

% Reduction in avg FCT

Flow Size

- InitCWnd = 10
- InitCWnd = 50
- RC3

Flow Size:
- 1.5KB
- 3KB
- 4KB
- 7.5KB
- 10KB
- 60KB
- 100KB
- 200KB
- 400KB
- 1.5MB
- 3MB
Comparison: Increasing InitCwnd

Short Flows penalized by Increasing Initial Congestion Window; Benefit from RC3
Comparison: Increasing InitCwnd

Long flows see stronger gains with RC3.
RC3 in comparison

- Increasing the initial congestion window

- Rate Control Protocol (RCP)
Comparison: RCP

![Graph showing the percentage reduction in avg FCT for different flow sizes comparing RCP and RC3. The x-axis represents flow size in KB and MB, and the y-axis represents the percentage reduction in avg FCT. The graph shows that RCP generally outperforms RC3 across all flow sizes.](image)

- **Red Bar** represents RCP
- **Gray Bar** represents RC3
Comparison: RCP

RCP penalizes short flows due to more aggressive long flows and explicit pacing.
Long flows see similar performance gains
Stress Testing RC3

- Varying Link Utilization
- Varying RTTxBW
- More Topologies
- Different Workload
- Link Heterogeneity
- Random Losses
- Varying Priority Assignments
- Application Pacing
- Comparison with traditional QoS
Roadmap

• *Isn’t congestion control a solved problem?*
  – Conflicting goals of high throughput and friendliness decoupled through priorities

• **Scope for performance gains**
  – *Increases with increasing RTTxBW*

• **Design Details**
  – Additional packets sent backwards from the end using multiple low priority levels

• **Simulation Results**
  – 40-80% reduction in FCT over baseline TCP implementation

• **Linux Implementation and Evaluation**

• *Challenges and Future*
RC3 in Implementation

• Implemented in Linux 3.2 kernel

• 121 additional LOC
  – Sending Data Packets: 74 LOC
  – Receiving Data Packets and Acks: 47 LOC

• Agnostic to the underlying TCP algorithm
  – Can be Tahoe, Reno, NewReno, BIC, CUBIC etc
Evaluation

**Average FCT (secs)**

- **Regular TCP**
- **RC3**

**Flow Size**

- 1.5KB
- 7.5KB
- 15KB
- 75KB
- 150KB
- 750KB
- 1.5MB
Evaluation

Low priority out-of-order packets processed by slow path
High per-packet CPU overhead
Leveraging NIC Offloading

• TCP Segmentation Offload (TSO)
  – Multiple segments processed by sender stack as a single chunk

• Large Receive Offload (LRO)
  – Multiple segments received aggregated into a single chunk

• RC3 supports offloading to reduce CPU overhead
  – Logically treat each chunk as a single packet at the sender
  – This allows aggregation of segments at the receiver
Leveraging NIC Offloading

The graph shows the average FCT (seconds) for different configurations:

- No TSO/LRO
- TSO
- LRO
- Both TSO/LRO

The y-axis represents the Average FCT in seconds, ranging from 0.000 to 0.018.
Isn’t congestion control a solved problem?
   - Conflicting goals of high throughput and friendliness decoupled through priorities

Scope for performance gains
   - Increases with increasing RTTxBW

Design Details
   - Additional packets sent backwards from the end using multiple low priority levels

Simulation Results
   - 40-80% reduction in FCT over baseline TCP implementation

Linux Implementation and Evaluation
   - Simple modifications, agnostic to the underlying congestion control algorithm

Challenges and Future
Where RC3 is of little help...

- Low delay bandwidth product
- Very heavily utilized links
- Small queue buffer size at the bottleneck
- Application pacing
Deployment Concerns

• Partial Priorities Support

• Middleboxes [Honda et. al. 2011, Flach et al. 2013]

• Wireless
Future

- Performance gains increase with BWxRTT
  - Likely to increase with time

- Futuristic datacenter bandwidth of 100Gbps
  - 45% reduction in average FCT (over flows)
  - 66% reduction in average FCT (over bytes)
Summary

- Send additional packets from a flow using several layers of low priority service

- Uses only the spare capacity in the network without affecting the regular traffic

- Gives 40-80% reduction in FCTs over baseline TCP