

Dynamic MultiPath Routing

Network Working Group Internet-Draft

By

F. Devetak and S. Kapoor

Illinois Institute of Technology

Motivation

- Internet Traffic is growing
 - New applications and services added constantly
- Network Throughput must be maximized
 - Multipath Routing could use more elasticity
 - Manage congestion effectively
 - Address fair allocation of resources
 - Preventing bottlenecks

Multipath Routing

- ECMP and MPTCP
 - Choice of paths independent of dynamic network conditions
 - ECMP
 - only equal cost paths
 - MPTCP
 - predetermined set of paths
 - not Pareto-Optimal

Dynamic Multipath Routing

- Dynamic Multipath Routing
 - New paths discovered dynamically
 - Improved Throughput
 - Congestion Management
 - Fairness amongst different source-destination pairs

Two Methods

- First Method: Congestion Control
 - Dynamic Re-Routing of Flows based on network conditions
- Second Method: Congestion Control with Fairness
 - Dynamic Re-routing of Flows based on network conditions
 - Fairness Enforcement
- Both Methods use
 - Additive Increase
 - Multiplicative Decrease
 - Combination of source routing and Link-State routing (OSPF)
 - Link-State Advertisements with queuing delay and queue occupancy

Method 1 (Congestion Control)

- Congestion Control
 - Based on congestion of a path's links
 - Expressed as variable $b_i = [0,1]$
- Flow on Path i at time t :
 - Multiplicative Decrease based on congestion
 - $x_i(t) = (1-b_i) * x_i(t-T)$ [T=cycle time]
 - Shortest Path based on link delay
 - If Shortest Path is new, add to path list
 - Additive Increase on shortest path [to achieve and maintain demand]
 - $x_i(t) = a + (1-b_i) * x_i(t-T)$

Method 2

(Congestion Control with Fairness)

• Congestion Control and Fairness Function

– Expressed as function of PBF

– PBF as b_i in Method 1

– $PBF = \frac{\text{Total Flow}}{\gamma * \text{Demand}}$ [γ = fairness coefficient]

• Flow on Path at time t:

– $x_i(t) = (1 - b_i) * x_i(t - T)$

– Shortest Path based on link delay

– If Shortest Path is new, add to path list

– Additive Increase on Shortest Path [to achieve and maintain demand]

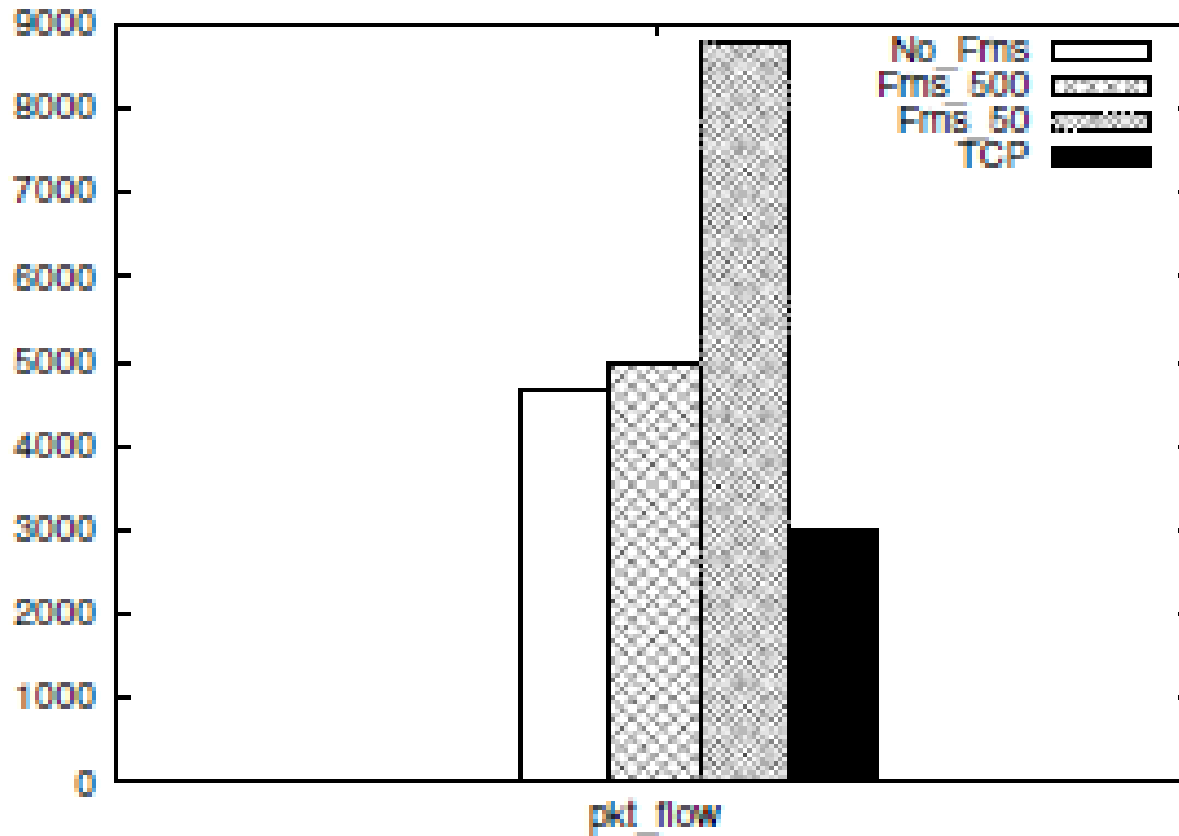
• $x_i(t) = a + (1 - b_i) * x_i(t - T)$

• $x_i(t) = a + (1 - b_i) * x_i(t - T)$

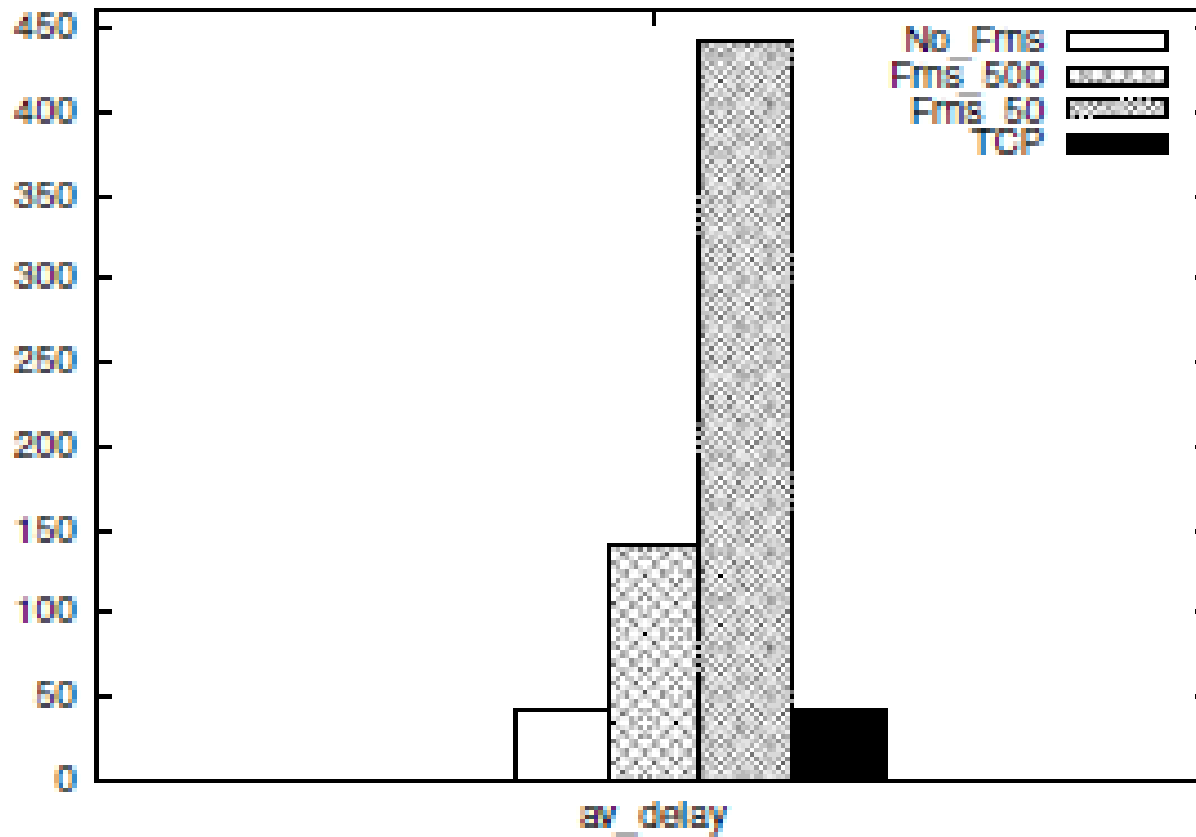
Implementation Details

- Both methods use factor b_i for flow control, but calculation is different
 - Method 1: b_i based on link queue occupancy
 - Method 2: b_i combination of method 1 and fairness factor
- Factor a is a unit of flow increase (integer)
- Source Routing using Nix-Vector Routing (NS-3)
- Shortest paths found using OSPF (NS-3) with some modifications
 - Link Cost = queuing delay
 - Queue occupancy of O/G link added to LSA
- Simulations:
 - multiple users starting at random times
 - background traffic

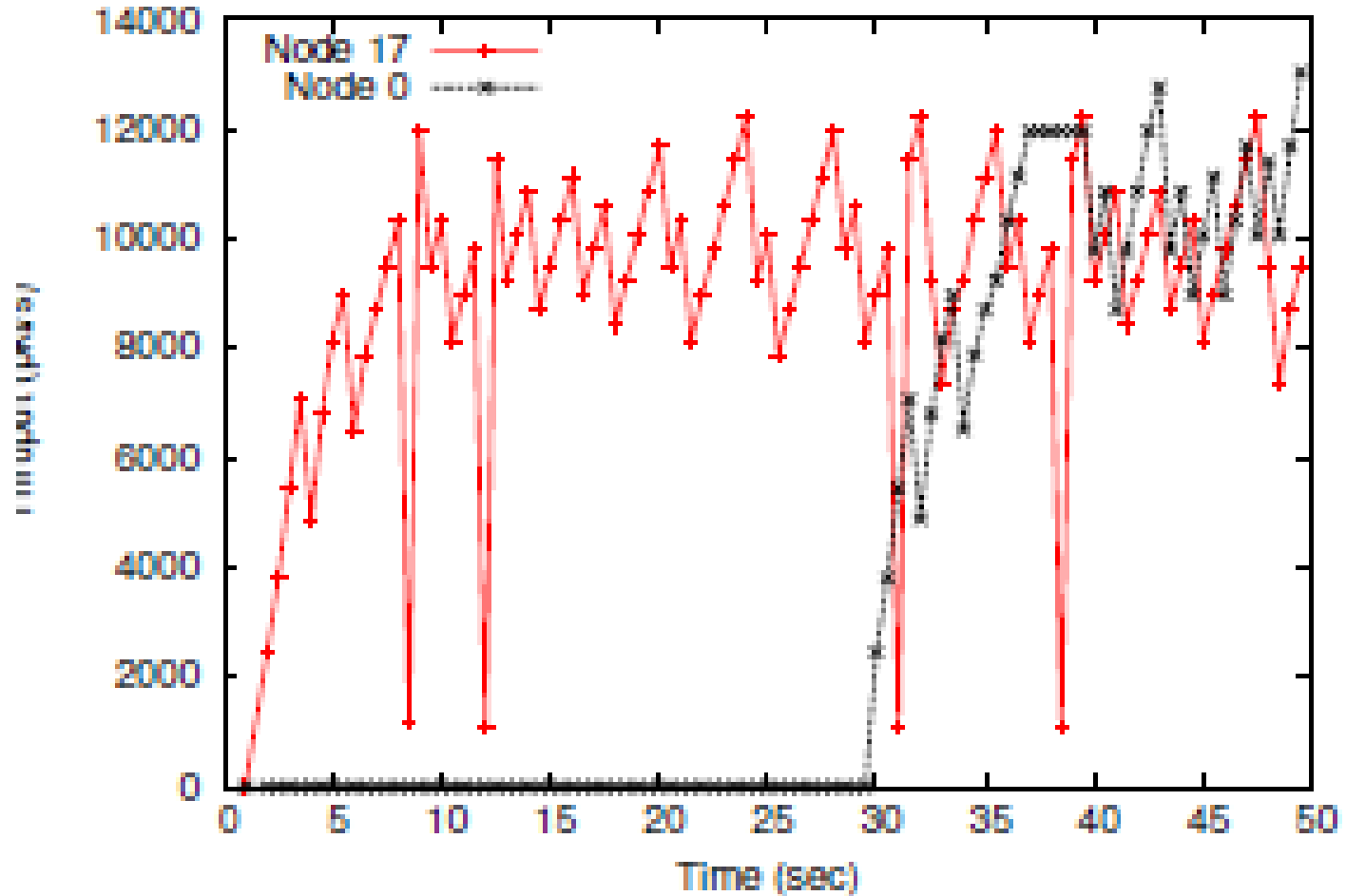
NS3 Simulations I



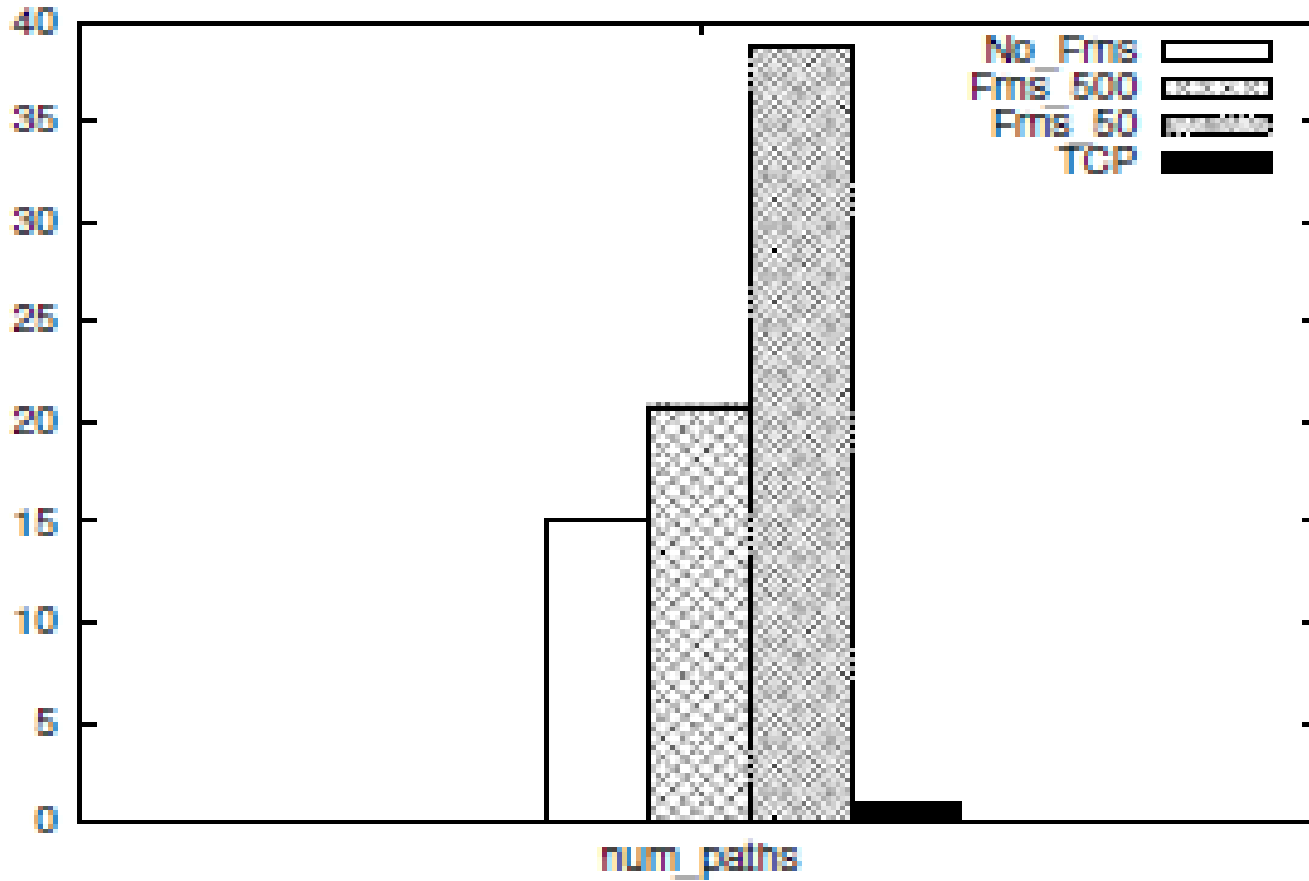
NS3 Simulations II



NS3 Simulations III



NS3 Simulations IV



Conclusion

- Improved Throughput
- Improved Fairness
- Further studies needed
 - Total Network Throughput
 - Reduce number of paths
 - Packet acknowledgement
 - Best method for Source Routing