

Project

Network Simulator 2

Part 1: TCP Flows and Congestion Control, Performance Study

1.1

Define a graph with eight nodes, including one core link acting as a **bottleneck**, and six leaf nodes (three leaves on each side of the core link). Regarding the traffic: you will establish two flows for each emitter/receiver pair. The emitters will be positioned on one side of the core link and the receivers on the other. Each of your six flows will be associated with a **FTP-type application using TCP** at the transport layer. Ensure that the **RTTs of the paths used for each emitter/receiver pair** are sufficiently heterogeneous to carry out your first experiments on **TCP fairness with respect to RTT**.

1.2

Generate congestion on the core link over a period long enough to highlight **significant results** (whether for RTTs or other original analyses). Analyze the behavior of each flow and discuss the **fairness of bottleneck sharing** as a function of the latency introduced on each path. Without congestion on the core link, what do you observe when the **reception window** is not the factor limiting the flows throughput?

1.3

Analyze the **evolution of the load on the bottleneck link over time** by varying your simulation parameters (particularly the load on the waiting queue). Use different **TCP versions** and compare them (at this stage, its preferable to revert to homogeneous delays to avoid crossed and multi-criteria experiments). Modify the nature of the TCP emitters/receivers (e.g., **NewReno**, **Cubic**, **Vegas**, **Sack**, **Delayed Ack**, etc.) as well as the configuration of their internal parameters, with the goal of improving performance particularly the throughput of the flows (compared to an ideal case used as a reference) and **fairness** (with respect to latency and RTT).

Analyze **competition (aggressiveness, fairness)** between different TCP flows when their characteristics are identical and when they differ.

1.4

Modify the **size and possibly the nature of the queue** at the core link (within a sufficiently large interval to draw meaningful conclusions) and discuss the new performance results obtained. Do you observe significant differences? Try to interpret them.

1.5

Show that certain versions of TCP are **more scalable** than others for achieving high data rates (10 Gbps). Then point out the potential **drawbacks** of these variants.

1.6

Comment on and analyze in detail the results obtained when your network load increases (at least on the core link that you will deliberately overload with background traffic and/or additional TCP flows): its load, its queue, and the **useful throughput** (i.e., effective throughput) of the flows crossing it. Perform a specific analysis for the **worst-performing flows**, i.e., those experiencing the most retransmissions (such that their useful throughput is much lower than their measured raw throughput). These will serve as a **reference set of TCP flows**, illustrating behavioral differences among them and possibly among different TCP versions.

Make use of the results obtained in the previous exercises (whether related to queue characteristics or TCP variants), and remember to **evaluate your simulations over different time scales** to better understand the nature of the congestions caused by the additional injected traffic.

Part 2: Original Analysis

Propose an **original analysis** of your choice and try to highlight a **lesser-known phenomenon** than those you have illustrated in the previous questions (related to TCP and/or its interactions with queues).

The basis for the evaluation of this second part will be a **detailed and illustrated report**, commenting on and interpreting your results (maxi-

imum 15 pages, excluding configuration appendices). Your configurations should also be commented on in detail to justify your choices.

Provide your **configuration scripts** and justify all your **network choices** (statistical distributions, queue size/type, type of flow, TCP configuration, etc.) in terms of realism. Also justify the **evaluation criteria** used for your comparisons. The use of **statistical methods** (at least confidence intervals) for processing data from your simulations is strongly recommended. Take care with your **graphical representations**, ensuring they include as much explanation and interpretation as possible.

Your **summary report** should first clearly and precisely describe your **simulation environment** (network parameters used and relevant for the transport layer) to justify the soundness and relevance of your analyses. Your comparisons should be **reasonable** and incremental for example, **single-criterion** at least initially. Then, the main goal will be to **analyze, compare, and comment** on the behavior of your simulations both at the **link/queue level** (operators perspective) and at the **TCP flow level** (application perspective for users).

Highlight the **limitations** of your simulation choices, both from the operators side (link load) and from the application side (useful throughput). In fact, be sure to consider **useful throughput rather than raw throughput** in your advanced analyses.

Part 3: NS2 Development and ECMP Extension

By default, NS2 supports multi-path routing of the **ECMP (Equal-Cost Multi-Path)** type. However, packet distribution is performed in a very simplistic **round-robin** mode. Here, we will improve this aspect by proposing a **flow-based distribution** instead.

To do so, you will need to **modify the routing parameters** in the file `multipathclassifier.cc`, and then, in a second step, also **mark your flows via the TCP agents** (this means you will have to modify both **C code** and **OTCL code** to interact from the TCL interface).

3.1

Observe, in a scenario of your choice, how ECMP in round-robin mode can degrade the performance of TCP flows.

3.2

Implement a **flow-based distribution** and show that it solves the issues identified in the previous question.

3.3

Despite this improvement, does ECMP **necessarily provide better performance** than single-path routing? If not, find and demonstrate a **counterexample**.

Summarize all your most relevant results and their interpretations in a **short report** of no more than **10 pages** (excluding configuration appendices). Provide your **configuration scripts** and the **NS2 code** you have modified.