Technical Disclosure Commons

Defensive Publications Series

November 26, 2018

Using Propagation Delay in Weighted Decision Routing Algorithms

Nicholas George McDonald
Hewlett Packard Enterprise

Follow this and additional works at: https://www.tdcommons.org/dpubs_series

Recommended Citation
McDonald, Nicholas George, "Using Propagation Delay in Weighted Decision Routing Algorithms", Technical Disclosure Commons, (November 26, 2018)
https://www.tdcommons.org/dpubs_series/1698

This work is licensed under a Creative Commons Attribution 4.0 License.
This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.

Using Propagation Delay in Weighted Decision Routing Algorithms

Abstract

Routing algorithms that use a weighted decision based on queue occupancy and hop count (e.g., UGAL) estimate the expected remaining delay by multiplying the hop count and queue occupancy. They choose the route that yields the lowest weight which approximates the lowest delay. This algorithm assumes all channel lengths are equal, which is very untrue in large scale networks. This invention integrates the propagation delay into the equation to yield a realistic expected delay value to be compared with other options.

Background

Many adaptive routing algorithms follow the UGAL methodology where minimal and non-minimal routes are compared using weights. UGAL is a source adaptive mechanism but the methodology also works for incremental adaptive routing. This methodology was first described as follows:

$$weight_{min} = congestion_{min} * hopcount_{min}$$

$$weight_{nonmin} = congestion_{nonmin} * hopcount_{nonmin}$$

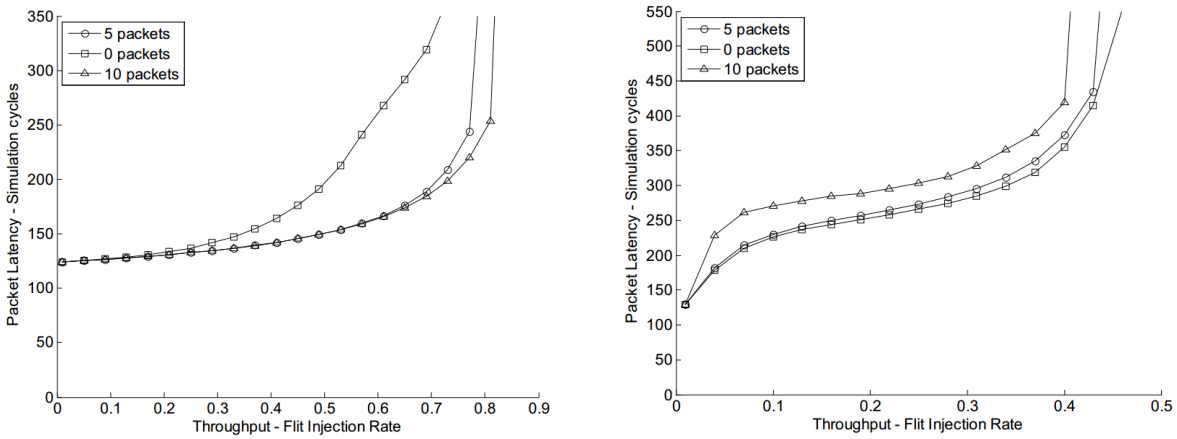
The methodology chooses the route with the lowest weight. This methodology was developed for systems where the channel latencies were very small. With the invention of high-radix routers, channel latency has become a significant contribution to latency and has presented significant problems due to credit round trip times for credit-based flow control.

Several people have included a fixed bias value to the non-minimal route weights to attempt to overcome these issues. This is done as follows:

$$weight_{nonmin} = congestion_{nonmin} * hopcount_{nonmin} + bias$$

Adding a fixed bias value to the non-minimal route weights makes the non-minimal routes less desirable. The ramification of this is that non-minimal routing will be chosen less which generally helps load-balanced traffic patterns (e.g., uniform random) but significantly hurts traffic patterns that are not load-balanced.

The following two figures were produced by Jiang et al. in 2009 when he introduced the bias in attempts to solve the perceived issues. The first figure shows the effects of using a fixed bias on uniform random (load-balanced) traffic. As shown, a higher bias value is better because it reduces the non-minimal routes. The second figure shows the effects of using a fixed bias on worst cast (non-load balanced) traffic. In contrast, a lower bias is better because it encourages non-minimal routes that are crucial for load-balancing the non-load balanced traffic.



(a) Threshold induced variations – PB on UR traffic (b) Threshold induced variations – PB on WC traffic

Problem Statement

Weighted decision routing algorithms don't consider propagation delay in their calculations. This can lead to unnecessary high latencies because packets may cross long latency links when low latency links are available. Consider the case where a routing algorithm is deciding between two paths with 2 hops each. They both have the same queue occupancy and hop count. One path has two short channels and the other path has two very long channels. The standard mechanism for computing an estimated delay would equally choose between these two paths. This yields unneeded high latency transactions on the network. All modern high performance low diameter networks (e.g., HyperX, Dragonfly, Slimfly, etc.) have channel lengths that can vary by multiple orders of magnitude.

Methodology

Routing tables hold information on a per destination basis. In order to determine out which output port the packet should take, the routing table is indexed by the destination's identifier. The output of the table is a list of routes that are available to the packet. Under the baseline design (e.g., like in a UGAL implementation) each route entry would list the egress port and the number of hops to the destination if that egress port was used. Using congestion information from the output ports, the algorithm multiplies the hop count and queue occupancy then selects the output port with the lowest expected latency. This only estimates the delay with respect to the queues, not the channels.

This invention specifies that the expected latency computation include propagation delay by multiplying the queue occupancy by the hop count then adding the propagation delay from the current location to the destination along the path selected. This style of delay estimation includes the queues and the channels.

For high performance networks that exist within a single administrative domain, the propagation delay from any location to another location is explicitly known as the cable lengths can be known explicitly or implicitly. While academic simulations used fixed length channels,

this is highly unrealistic to real-world systems. This is especially true of low diameter networks such as HyperX, Dragonfly, and Slimfly. The propagation delays along different paths can vary by multiple orders of magnitude (1ns to 500ns).

Advantages

The quantitative benefits of this invention depend highly upon the system configuration and the chosen routing algorithm. For the HyperX, some dimensions will have very short latencies (1-10ns) and other dimensions will have long latencies (100s of ns). The HyperX often yields higher relative bisection bandwidth in some dimensions than others. With these two insights combined with using propagation delays in the routing algorithm, systems can be constructed with excess bisection bandwidth in the dimensions with low latencies. This will allow the routing algorithm to utilize each dimension proportionally to its relative bandwidth and latency. This allows non-minimal network load-balancing without taking a huge latency hit. For the baseline system you would expect any additional hop in the network to be the average of all channel latencies (~100ns). With the propagation delay included in the routing algorithm, you can expect an additional hop to be closer to the minimum channel latency (~5ns).