A Scalable Content-Addressable Network

S. Ratnasamy, P. Francis, M. Handley, R. Karp,
S. Shenker

Proceedings of ACM SIGCOMM ’01

Sections : 3.8, 4

Πλίτσης Ζήσης
Ρόβα Ευθυμία

Caching and Replication

Caching

- In addition to its primary data store (*i.e.* those data keys that hash into its coordinate zone), a CAN node maintains a cache of the data keys it recently accessed.
- Thus, the number of caches from which a data key can be served grows in direct proportion to its popularity and the very act of requesting a data key makes it more widely available.

Replication

- A node that finds it is being overloaded by requests for a particular data key can replicate the data key at each of its neighboring nodes.
- A popular data key is eventually replicated within a region surrounding the original storage node, causing the load to be spread over the entire region.

Cached and replicated data keys should have an associated *time-to-live field* and be eventually expired from the cache.

Design Review

Metrics

Metrics

- **Path length**: the number of (application-level) hops required to route between two points in the coordinate space.
- **Neighbor-state**: the number of CAN nodes for which an individual node must retain state.
- **Latency**: we consider both the end-to-end latency of the total routing path between two points in the coordinate space and the per-hop latency, i.e., latency of individual application level hops obtained by dividing the end-to-end latency by the path length.
- **Volume**: the volume of the zone to which a node is assigned, that is indicative of the request and storage load a node must handle.
- **Routing fault tolerance**: the availability of multiple paths between two points in the CAN.
- **Hash table availability**: adequate replication of a (key,value) entry to withstand the loss of one or more replicas.

Parameters

Design Parameters

- dimensionality of the virtual coordinate space: *d*
- number of realities: *r*
- number of peer nodes per zone: *p*
- number of hash functions (*i.e.* number of points per reality at which a (key,value) pair is stored): *k*
- use of the RTT-weighted routing metric
- use of the uniform partitioning feature (Section 3.7)

Effect of design parameters on performance metrics

Design parameters	Path length (hops)	Neighbor state	Total path latency	Per-hop latency	Size of data store	Routing fault tolerance	Data store availability
Dimensions:d	$O(dn^{1/d})$ (fig:4)	$O(d)$	↓ (due to reduced path length)	-	-	↑	-
Realities:r	↓ (fig:5)	$O(r)$	↓ (due to reduced path length)	-	$O(r)$	↑	$O(r)$
Number of peer nodes per zone:p	$O(1/p)$	$O(p)$	↓ (due to reduced path length and reduced per hop latency)	↓ (table:2)	Replicated data store: O(p), partitioned data store: -	↑ (due to backup neighbors)	Replicated data store: O(p), partitioned data store: -
Number of hash functions:k	-	-	↓ (fig:7)	-	$O(k)$	-	$O(k)$
Use of RTT-weighted routing metric	-	-	↓ (due to reduced per hop latency)	↓ (table:1)	-	-	-
Use of uniform partitioning feature	Reduced variance	Reduced variance	-	-	Reduced variance (fig:9)	-	-

Simulation

To measure the cumulative effect of all the above features, we selected a system size of $n=2^{18}$ nodes and compared two algorithms:

1. a “bare bones” CAN that does not utilize most of our additional design features
2. a “knobs-on-full” CAN making full use of our added features (without the landmark ordering feature from Section 3.7)

- Simulation of Transit-Stub topology $n = 2^{18}$ (256k nodes)

Parameter	"bare bones" CAN	"knobs on full" CAN
d	2	10
r	1	1
p	0	4
k	1	1
RTT weighted routing metric	OFF	ON
Uniform partitioning	OFF	ON
Landmark ordering	OFF	OFF

Metric	"bare bones" CAN	"knobs on full CAN"
path length	198.0	5.0
# neighbors	4.57	27.1
# peers	0	2.95
IP latency	115.9ms	82.4ms
CAN path latency	23,008ms	135.29ms

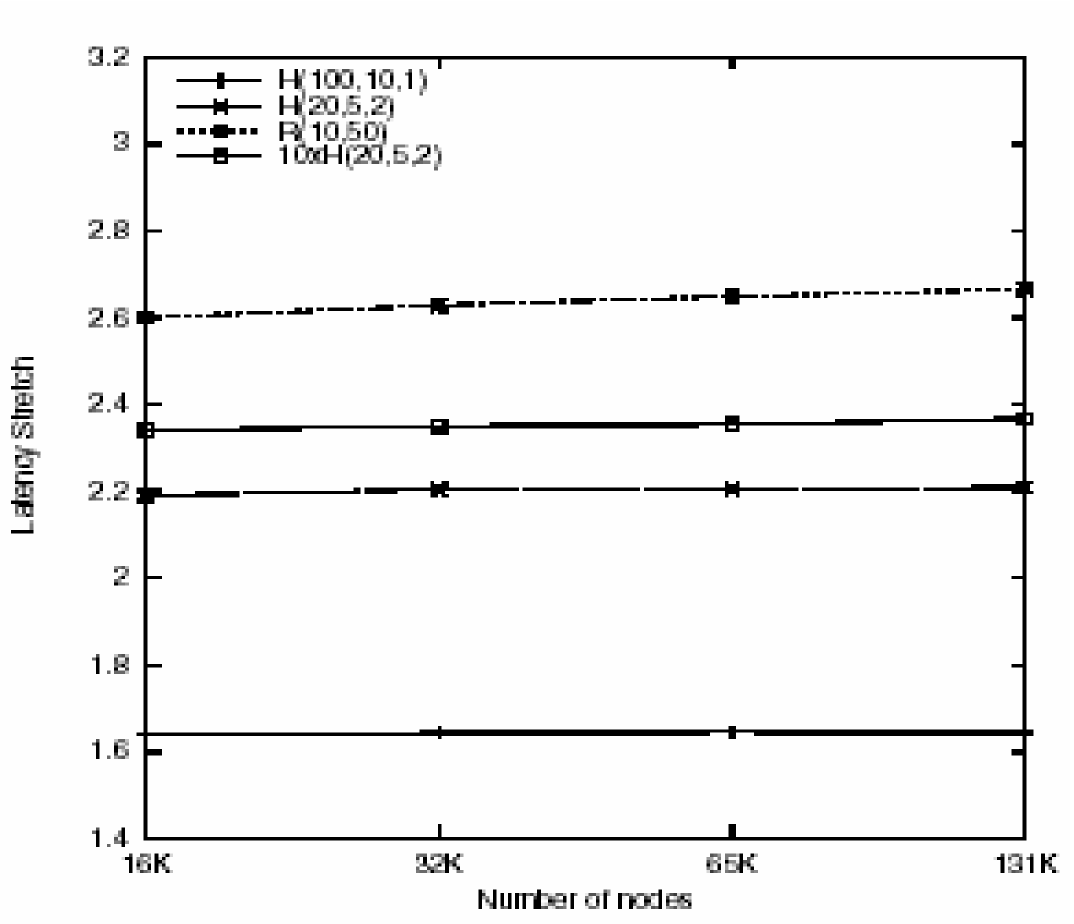
The effect of Link Delay Distributions

- H(100; 10; 1): A Transit-Stub topology with a hierarchical link delay assignment of 100ms on intra-transit links, 10ms on transit-stub links and 1ms on intra-stub links. This is the topology used in the above "knobs-on-full" test.
- H(20; 5; 2)
- R(10; 50): A Transit-Stub topology with the delay of every link set to a random value between 10ms to 50ms.
- 10xH(20; 5; 2): The same as H(20; 5; 2) except that the density of CAN nodes on the resultant topology is about 10 times lower.

The effect of Link Delay Distributions

- The latency stretch: the ratio of CAN latency to IP latency, was measured for different system sizes.
- While the delay distribution affects the absolute value of the latency stretch, in all cases, the latency stretch grows very slowly with system size. In no case do we see a latency stretch of more than 3 for system sizes

The effect of Link Delay Distributions



Ερωτήσεις

?

Example of internet domain structure