Weighted Highest Random Weight (HRW) and its Applications

Satya R Mohanty
Mankamana Misra
Ali Sajassi
Acee Lindem
IETF 104 Prague
The Load Balancing problem

Given a set of objects and servers, devise a mapping of objects to servers that ensures uniform load balancing and minimal disruption due to reassignments.

Modulo-N Assignment: $S = \text{key}\%N$

When one server goes down or comes up, a lot of reassignments!
Score = Hash(Srvr-id ⊕ Key)
Highest score wins!

Object j assigned to Server i when H(Si ⊕ Oj) is highest

Weighted HRW Problem

• **What happens** when the Servers are **not** of equal capacities or weights?

• One approach: Take the weighted score:
  \[ f_i \cdot \text{Hash}(\text{Srvr-id} \cdot \text{Key}); \text{ where } f_i = \frac{w_i}{\sum w_j}, \ j=1,..,N; \]
  
  \( N \) is number of Objects

• Does it obey HRW properties?
Score = $f_i \times \text{Hash(Srvr-id * Key)}$

Highest score wins

Overall Computation is: $O(#\text{Srvr} \times #\text{Objects})$

Computation for objects 597 and 318 not shown for brevity
\[ \text{Score} = f_i \times \text{Hash(Srvr-id \times \text{Key})} \]

Highest score wins

- Weight of S2 only changed.
- But load factors changed everywhere!
- Results in **re-computation** and may **re-assign** in a potentially disruptive manner.
- Overall **re-computation** is \( O(\#Srvrs \times \#Objects) \)
- Does not satisfy HRW desirable properties

Computation for objects 597 and 318 not shown for brevity
Weighted HRW Solution

• Conclude that weighted score is not efficient
  \[ f_i * \text{Hash(Srvr-id * Key)}; \text{ where } f_i \text{ is } w_i/\text{sum}(w_j), j=1,\ldots, N \]

• Take the score as: \[-w_i/\ln(\text{Hash(Srvr-id * Key)}/H_{\text{max}})\]


- Need to re-compute the score for only the server whose weight changed. Other’s scores do not change. Order is \( O(\#\text{Objects}) \).

- Obeys the minimal disruption properties of the HRW
  • When a server is added/removed or changed, only the scores for that node change.
  • It may win some keys (if score increases)
  • It may lose some keys (if score decreases)
  • And it does so with minimal disruption
EVPN DF Election in A/A Deployments with DMZ link bandwidth


- Goal is to have different DFs (PEs) for different EVI (vi) for load balancing taking into account LB
- When LB changes we should have minimal number of reassignments
- Note that this reduces to the WHRW problem with the PE's ip-address as the srv-id, vlan-id (vi) as the object id, and link-bw as the weights!
Other Applications

• Resilient Hashing
  – Unequal cost multipath
  – LAG

• Multicast
  – Unequal B/W towards receivers
  – DR elections when access bandwidth is different for attach points in the last hop network
Thanks!!!
Highest Random Weight (HRW)

• When the hash function is uniform (any good hash function should satisfy this) and as the load (number of objects) increases, It is proved† that

  – The load is evenly balanced across the servers using HRW
  – Minimal disruption property: a server going up or down results in a minimal reassignment of impacted objects

†Using name-based mappings to increase hit rates: Thaler et. al. IEEE Transactions on Networking, 1999
Hash(Srvr-id * Key) = Score

Highest score wins

S3 goes down!

H(S1* 233) = 457
H(S1* 318) = 471
H(S1* 597) = 919

H(S2* 233) = 317
H(S2* 318) = 513
H(S2* 597) = 200

H(S3* 233) = 512
H(S3* 318) = 172
H(S3* 597) = 706
Hash(Srvr-id * Key) = Score
Highest score wins

S4 comes up!

H(S1* 233 ) = 457
H(S2* 233 ) = 317
H(S3* 233 ) = 512
H(S4* 233 ) = 236

H(S1* 318 ) = 471
H(S2* 318 ) = 513
H(S3* 318 ) = 172
H(S4* 318 ) = 672

H(S1* 597 ) = 919
H(S2* 597 ) = 200
H(S3* 597 ) = 706
H(S4* 597 ) = 234
Resilient Hashing

- Minimize flow remapping in Trunk/ECMP Groups in FIB
  - Many vendors.....
  - But nothing on UCMP?

Flows hashed on 5-tuple