CONSISTENT HASHING

USED BY PARALLEL CRAWLERS TO DIVIDE THE WEB BETWEEN MORE ENTITIES, IN ORDER TO AVOID DUPLICATION

1° approach

GIVEN D CRAWLERS, EACH URL IS HASHED WITH $H: URL \to \{0,\dots,D-1\}$, THEREFORE THE CRAWLER X MANAGES THE URLS U SUCH THAT $hash(U) = X$

↓ PROBLEM

IF WE DECREASE OR INCREASE THE NUMBER OF CRAWLERS, WE HAVE TO RECOMPUTE THE HASHES, IN ORDER TO RE-DISTRIBUTE THE WORKLOAD

↓ SOLUTION

CONSISTENT HASHING

(used by Chord !!)

How it works:

→ Items and crawlers are mapped to unit circle using an hash function ID()

→ The item K is assigned to the first crawler N such that $ID(N) \geq ID(K)$

Notes:

→ Usually each crawler is replicated across the circle $\log \beta$ times (SCALABLE!!!)

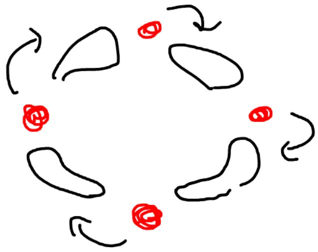
→ If a crawler N crashes, the crawler N' $(ID(N') > ID(N))$ inherits its items

→ If a new crawler N appears, the crawler N' $(ID(N') > ID(N))$ shares part of its items with him

→ Probability that an item goes to a crawler is $\leq \frac{O(1)}{\beta}$

#items??

→ Any crawler gets $\left(\frac{I}{\beta}\right) \log \beta$ items



EXERCISE

url_ID = {3, 4, 9, 2, 5, 7, 12, 11}

crawler_ID = {1, 2, 3}

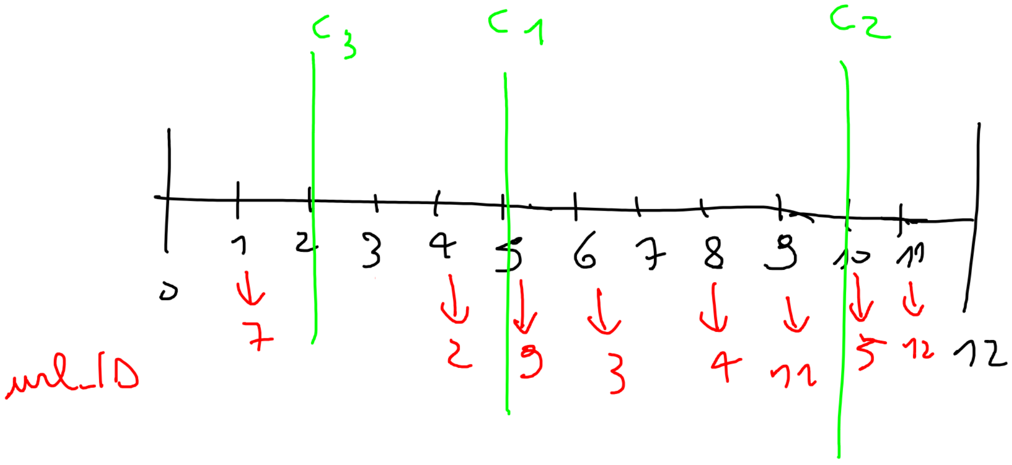
Use two hash functions, h_u e h_c, in the codomain $m = 13$

$$h_u(x) = 2x \mod 13$$

$$h_c(x) = 5x \mod 13$$

url_ID	h_u
3	6
4	8
9	5
2	4
5	10
7	1
12	11
11	9

crawler_ID	h_c
1	5
2	10
3	2



$C_1 = \{9, 3, 4\}$

$C_2 = \{5, 12, 7, 11\}$

$C_3 = \{2\}$