19/01/2019

Information Retrieval

- Esercizio 2
- Rsync F_old = "ABRACADABRA" (client) B = 3
  F_new = "ABRACALABRIA" (server)
- Partendo dal file F_old sul client, suddivido il file da aggiornare in blocchi di 3 char:
  h_1 = "ABR" h_2 = "ACA" h_3 = "DAB" h_4 = "RA$"
- I blocchi $\{h_1, h_2, h_3, h_4\}$ vengono inviati al server. Eseguendo uno scanning su F_new, si verifica quali elementi $\{h_1, .. h_4\}$ sono presenti in F_new. Se è presente un elemento h_i, viene saltato nello scanning sono saltati i caratteri dell'intero blocco e si continua lo scanning per i successivi caratteri

CLIENT SERVER

$\{h_1, h_2, h_3, h_4\}$ ⟶ ABRACALABRIA

h_1 h_2 h_1

- Viene generata una codifica di F_new generata dai blocchi trovati nello scanning di F_new e i caratteri mancanti.
  La codifica viene inviata dal server al client, così da ricostruire e aggiornare F_old

SERVER CLIENT

$h_1 h_2 L h_1 IA$ ⟶ Aggiorno F_old

- ~~Blocchi~~ Gli elementi $\{h_1, h_2, h_3, h_4\}$ sono funzioni hash dei caratteri nel blocco

- ZSYNC
- F_new sul server viene suddiviso in blocchi di dimensione = 3:

  F_new h_1 = "ABR" h_2 = "ACA" h_3 = "LAB" h_4 = "RIA"
- Viene eseguito un hashing sui blocchi come per rsync. Gli hash vengono inviati dal server al client.

  SERVER
  $\{h_1, h_2, h_3, h_4\}$ ——> CLIENT
- Viene eseguito uno scanning su F_old per identificare quali elementi di $\{h_1, h_2, h_3, h_4\}$ sono presenti. A differenza di rsync, lo scanning è eseguito carattere per carattere

  CLIENT
  "ABRACADABRA"
  h_1 h_2 h_1

  • Viene generata una bitmask per indicare quali h_i sono presenti in F_old

  b = 1 1 0 0
  1 2 3 4
- Il client invia la bitmask al server il quale esegue l'algoritmo di compressione degli elementi h_3 e h_4 mancanti, basandosi su h_1 e h_2 presenti

  CLIENT ——> SERVER
  1100 ——> h_1 h_2 | h_3 h_4

  ——> ABRACA | LABRIA

  <0, 0, <>
  <7, 3, 1>
  <4, 1, $>

  • h_3 e h_4 compressi vengono inviati al client che decomprime e ordina gli elementi così da ricostruire F_new

· ESERCIZIO 3

NODE	REFERENCE	OUTDEGR	COPYLIST	COPY BLOCK	EXTRA
14	0	9			A
15	1	10	CL	CB	E

↳ compressione basata su 1 NODO precedente (14)

A: extra nodes di 14 = LISTA DI 14

3, 10, 12, 13, 14, 17, 19, 21, 25

CL: copy list di 15, ovvero una bitmask con cardinalità = |LISTA 14| dove vengono settati a 1 gli elementi comuni delle liste di 14 e 15

14:	3		10		12	13	14	17	19	21		25	
15:	↓	5	10	11	12	↓	14	17	19	21	24	↓	33
			↓		↓	↓	↓	↓	↓	↓		↓	
CL =	0		1		1	0	1	1	1	1		0	

CL = 0 1 1 0 1 1 1 1 0

E: i nodi di 15 non in comune con la lista 14

E = {5, 11, 24, 33}

CB: copy block del nodo 15 rispetto alla sua COPYLIST.
Il primo bit è lo stesso della copy list. Successivamente si calcola la frequenza dei bit consecutivi uguali della copyLIST

	0	1 1	0	1 1 1 1	0
Freq:	1	2	1	4	1

· Il copyBLOCK CB è codificato con il primo bit = primo bit della copylist, seguito dalle FREQUENZE - 1

CB = 0 | (1-1) | (2-1) | (1-1) | (4-1) | (1-1)

primo bit

= 0 0 1 0 3 0 — ultimo bit: possibile ometterlo conoscendo la CARDINALITÀ (OUTDEGREE) della lista