Motori di Ricerca

presente e futuro prossimo

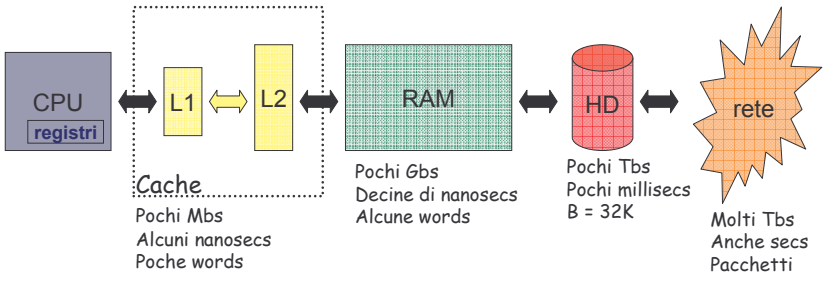
Cosa è un Compressore ?

Paolo Ferragina, Università di Pisa

Perché comprimere ?

Obiettivo: Eliminazione della ridondanza nei testi

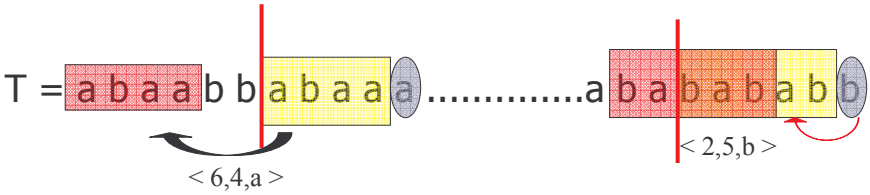
- Riduzione spazio
  - 33% tecniche standard (gzip, winzip,...)
  - 20% tecniche avanzate (bzip, ppm)
- Miglioramento delle prestazioni

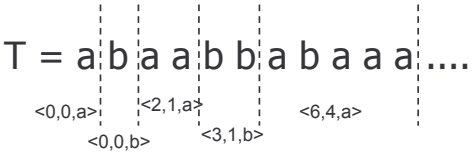


Paolo Ferragina, Università di Pisa

Gzip ('77-'78, raggiunge il 30%)

- Elimina ridondanza copiando pezzi di testo già visti





Più è lungo il testo, più ripetizioni ci aspettiamo, più risparmio otteniamo

Ogni sottostringa viene sostituita da una tripla <x,y,c> che rappresenta la distanza dalla sua precedente occorrenza (x), la sua lunghezza (y), e il carattere seguente (c).

Motori di Ricerca

presente e futuro prossimo

Cosa è una Lista Invertita ?

Indicizzatore: il "cuore" del motore

- Come organizzare l'informazione disponibile nell'archivio per rispondere velocemente alle interrogazioni poste dall'utente ?
  - Informazione = Pagine e hyperlinks, PS, PDF, PPT,…
  - Interrogazione = insieme di parole chiave

Es. Quali opere di Shakespeare contengono le parole (***Brutus*** *AND* ***Caesar*** AND *NOT* ***Calpurnia***) ?

Matrice binaria dei termini-documenti

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

1 se l'opera contiene la parola, 0 altrimenti

Per rispondere a Q=[***Brutus*** *AND* ***Caesar*** AND
- Prendiamo i vettori di ***Brutus, Caesar, Ca***
- Complementiamo il vettore di ***Calpurnia***
- Eseguiamo l'***AND*** logico bit-a-bit

Spazio occupato ?

- Consideriamo la seguente situazione:
  - Un milione di documenti
  - Ogni documento è di circa 6 Kb, e circa 1000 termini distinti
    - In totale abbiamo bisogno di 6Gb di dati
- Se il numero totale di termini distinti è 500,000 allora
  - La matrice [Term x Doc] è grande 500K x 1Mil = 500Gb

    …ma non più di 1M * 1,000 = 1Gb di uno

➢ Una migliore rappresentazione ?
  - ➢ più efficiente in spazio
  - ➢ che assegni una "rilevanza" a ogni risposta (documento)

Le Liste Invertite

Doc 2

So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious

Doc 1

I did enact Julius Caesar I was killed i' the Capitol; Brutus killed me.

I documenti sono analizzati per estrarre i termini e questi sono memorizzati insieme al corrispondente DocID

Term	DocID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Term	DocID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Ordiniamo tutti i termini in modo lessicografico per formare il Dizionario

Term	DocID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

Term	DocID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

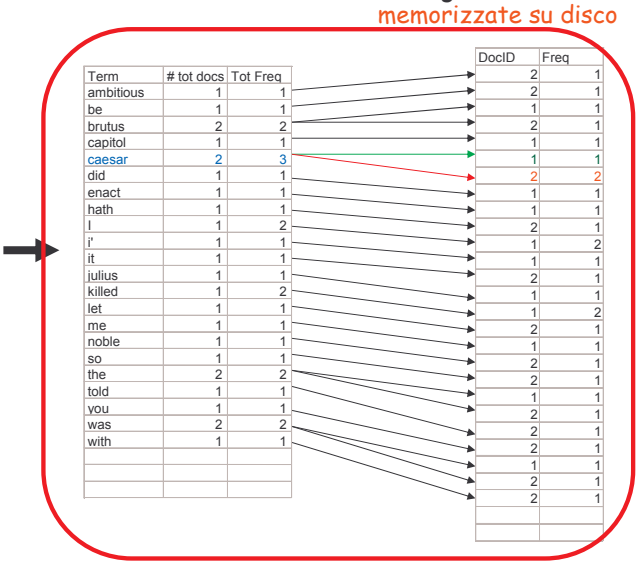
Le occorrenze dello stesso termine nello stesso documento sono “fuse insieme” incrementando opportunamente il valore della frequenza del termine

Term	DocID	Freq
ambitious	2	1
be	2	1
brutus	1	1
brutus	2	1
capitol	1	1
caesar	1	1
caesar	2	2
did	1	1
enact	1	1
hath	2	1
I	1	2
i'	1	1
it	2	1
julius	1	1
killed	1	2
let	2	1
me	1	1
noble	2	1
so	2	1
the	1	1
the	2	1
told	2	1
you	2	1
was	1	1
was	2	1
with	2	1

- Tutte le info suddivise in due file: *Dizionario* e *Posting list*

memorizzate su disco

Term	DocID	Freq
ambitious	2	1
be	2	1
brutus	1	1
brutus	2	1
capitol	1	1
caesar	1	1
caesar	2	2
did	1	1
enact	1	1
hath	2	1
I	1	2
i'	1	1
it	2	1
julius	1	1
killed	1	2
let	2	1
me	1	1
noble	2	1
so	2	1
the	1	1
the	2	1
told	2	1
you	2	1
was	1	1
was	2	1
with	2	1

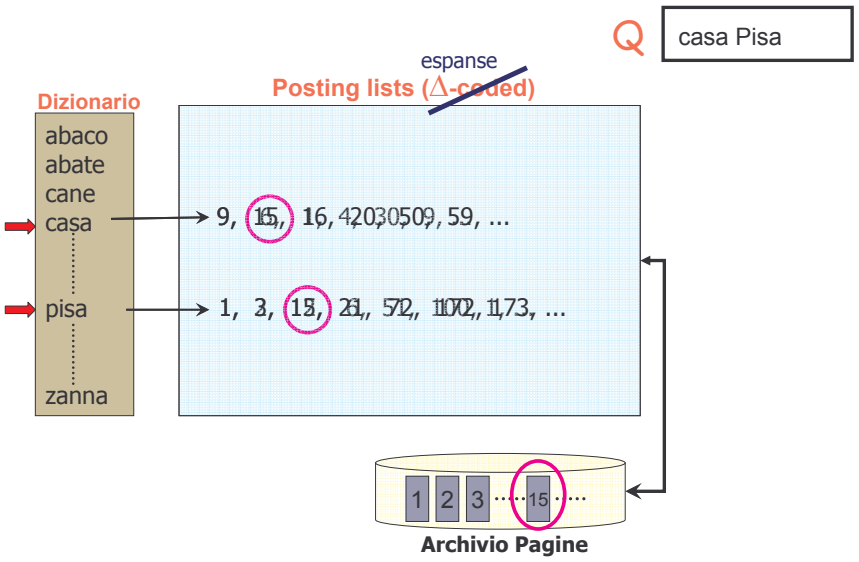


Memorizzazione delle *posting list*

- Un termine come ***Calpurnia*** occorre forse in un documento su un milione, e quindi possiamo codificarlo con molti bit ($\log_2$ 1M ~ 20).
- Un termine come ***the*** occorre probabilmente in ogni documento, quindi dovremmo usare pochi bit (20 sono troppi).
- Soluzione: ordiniamo la *posting list* di ogni termine per docID
  - ***Brutus***: 33, 47, 97, 107, 115, …
  - Δ : 33, 14, 50, 10, 8, …

…e speriamo che gran parte dei valori da rappresentare siano piccoli e richiedano dunque pochi bit.

Risoluzione di una interrogazione ?



Ricerca per Frase: 45% delle ricerche [Aprile 2003]

- Interrogazione ***"Paolino Paperino"***
- Non è più sufficiente la coppia <docID, freq>
- Ma occorre avere informazioni più dettagliate
  - <# *docs contenenti il termine corrispondente*;
    doc1: pos1, pos2, pos3, … ;
    doc2: pos1, pos2, pos3, … ;
    … >

Possiamo ancora comprimere la *posting lists*, agendo su *due livelli: documenti* e *posizioni*

Come risolvere una "phrase query"

- Si estraggono le posting list dei termini: ***Paolino, Paperino***
- Troviamo i documenti che contengono tutte le parole presenti nell'interrogazione
- Esaminiamo le posizioni delle occorrenze e garantiamo che siano consecutive nel documento
  - ***Paolino***:
    - *2*:1,17,74,222,551; *4*:8,27,101,429,433; *7*:13,23; ...
  - ***Paperino***:
    - *1*:17,19; *4*:17,191,291,430,436; *5*:14,19,101; ...