Intelligenza Artificiale

Anno accademico
2008-2009

Information Retrieval

Sommario

- Introduzione Information Retrieval
- Modelli di information Retrieval
  - Modello Booleano
  - Modello Vector Space Model (VSM)
  - Modello Statistico
  - Confronto tra modelli
- Ottimizzazione delle query
- Performance Evaluation

Introduzione (1/2)

- Information Retrieval: l'insieme delle tecniche utilizzate per il recupero mirato dell'informazione in formato elettronico. Per "informazione" si intendono tutti i documenti, i metadati, i file presenti all'interno di banche dati o nel www.
  - Possibilità di consultare le pratiche relative a casi in archivio in studi legali, assicurazioni, ma anche banche e aziende di serivizi.
  - Il termine fu coniato nel 1952 da Calvin Mooers che tra l'altro formulò la "legge di Mooers":
    - ***Un sistema di reperimento delle informazioni tenderà a non essere usato quando trovare le informazioni è "more painful and troublesome" che non trovarle.***

Introduzione (2/2)

- Information Retrieval è un campo interdisciplinare che nasce dall'incrocio di discipline diverse.
- Information Retrieval coinvolge la psicologia cognitiva, l'architettura informativa, la filosofia (vedi voce ontologia), il design, il comportamento umano sull'informazione, la linguistica, la semiotica, la scienza dell'informazione e l'informatica

Definizione

[Manning, Raghavan, Schütze - Introduction to Information Retrieval - 2008]

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collection (usually stored on computers).

Definizione

[Manning, Raghavan, Schütze - Introduction to Information Retrieval - 2008]

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collection (usually stored on computers).

Definizione

[Manning, Raghavan, Schütze - Introduction to Information Retrieval - 2008]

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collection (usually stored on computers).

Definizione

[Manning, Raghavan, Schütze - Introduction to Information Retrieval - 2008]

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collection (usually stored on computers).

Definizione

[Manning, Raghavan, Schütze - Introduction to Information Retrieval - 2008]

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collection (usually stored on computers).

Definizione

[Manning, Raghavan, Schütze - Introduction to Information Retrieval - 2008]

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collection (usually stored on computers).

Definizione

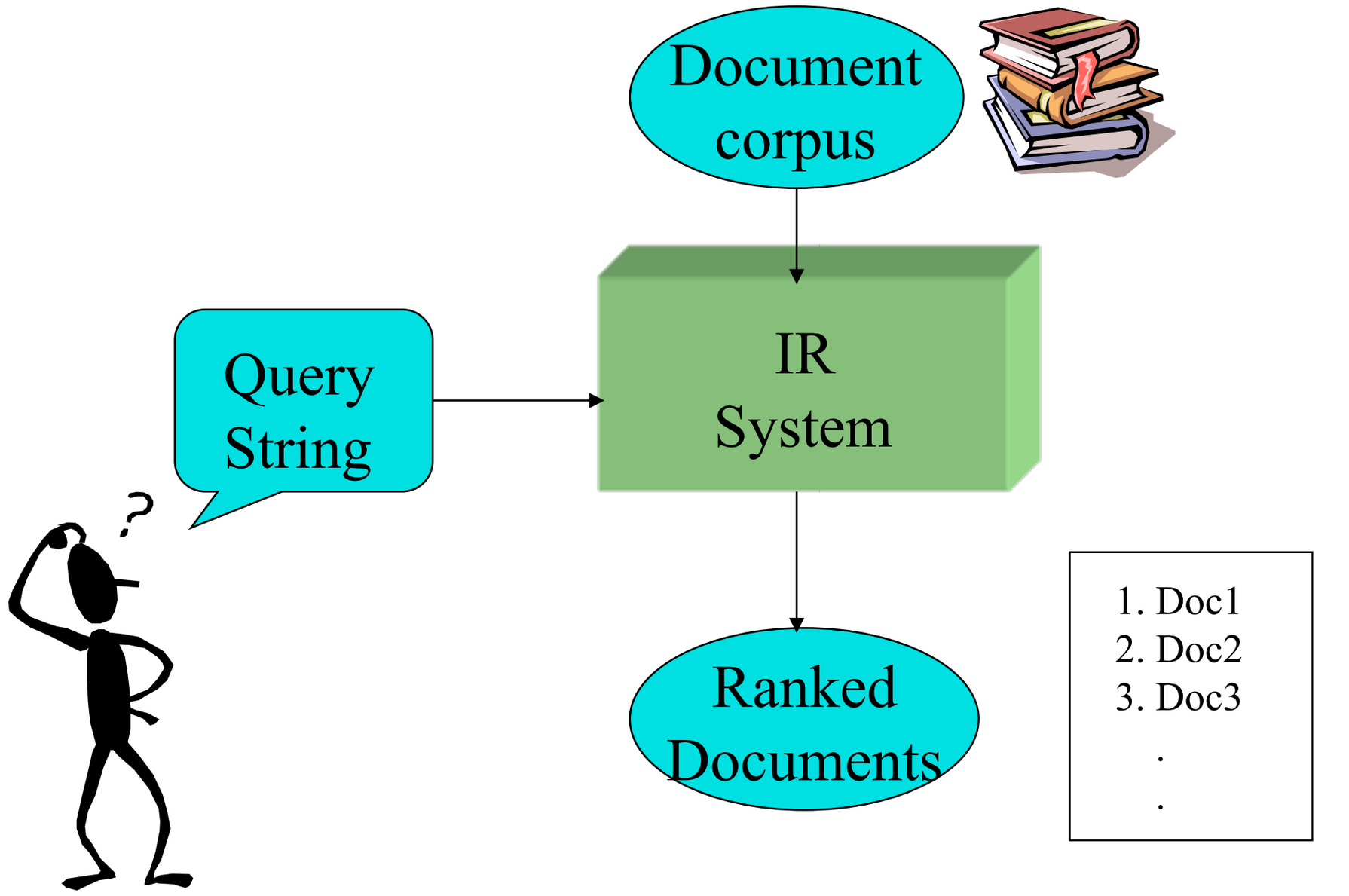
[Manning, Raghavan, Schütze - Introduction to Information Retrieval - 2008]

- Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from large collection (usually stored on computers).

Architettura IR (1/4)

- Per recuperare l'informazione, i sistemi di IR usano linguaggi di interrogazione basati su comandi testuali. Due concetti sono di fondamentale importanza, query ed oggetto:
  - Le query sono generalmente stringhe di parole-chiave rappresentanti l'informazione richiesta. Vengono digitate dall'utente in un sistema di IR
  - Un oggetto è un'entità che mantiene o racchiude informazioni in una banca dati. Un documento di testo, per esempio, è un oggetto di dati.
- A seguito di un'interrogazione, il sistema segnala il numero di documenti ritrovati e ordina i documenti per rilevanza.

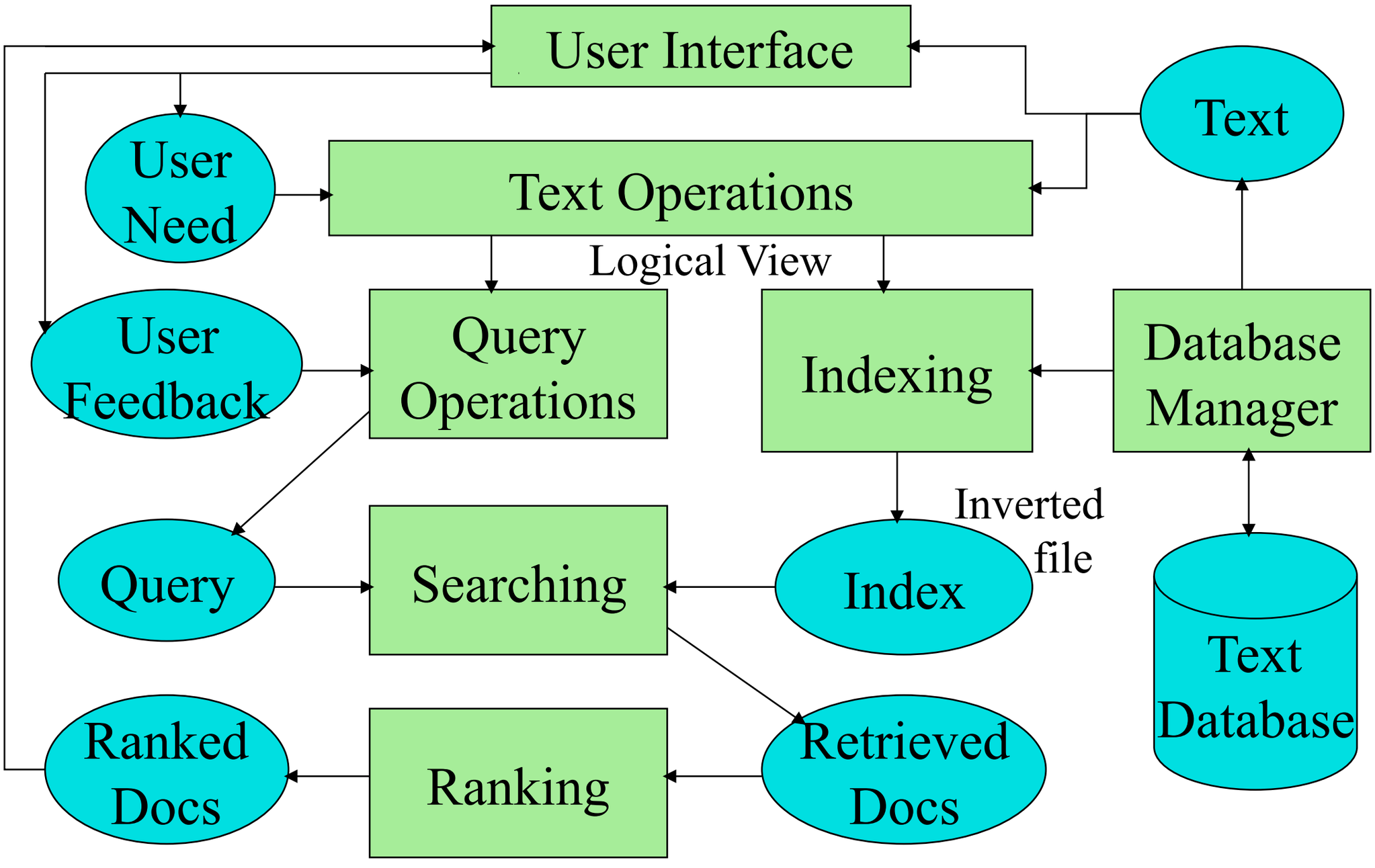
Architettura IR (2/4)



Architettura IR (3/4)

- L'esame dei documenti si avvale di due funzionalità:
  - **RANKING**: presentazione dei risultati in ordine crescente di rilevanza, in funzione dei pesi assegnati ai termini. L'utente può farsi un'idea di quanto la ricerca sia efficace.
  - **BROWSING**: documenti raggruppati in classi di somiglianza, permettendo all'utente di sfogliarli secondo un ordine logico.

Architettura IR (4/4)



Come ricercare?

- Nell'articolo The automatic creation of literature abstract, pubblicato nell'IBM Journal nell'aprile del 1958, Luhn affermava che:
  - la frequenza con cui alcune parole compaiono in un testo forniscono un parametro importante del significato delle parole
  - il posizionamento di queste parole all'interno delle frasi indica il significato e quindi l'importanza delle frasi
- La frequenza con cui alcune parole compaiono in un testo, può essere usata per rappresentare un documento.
- Questi saranno i principi di base all'indicizzazione automatica di testi.

Formalizzazione generale modello di IR

- Un modello di Information Retrieval è una quadrupla:
  $[D,Q,F,R(q_i,d_j)]$

1. D è un insieme composto di viste (o rappresentazioni) logiche di documenti della collezione

2. Q è un insieme di viste (o rappresentazioni) logiche delle query.

3. F (Framework) è l'insieme delle regole alla base delle rappresentazioni dei documenti e delle loro relazioni.

4. $R(q_i,d_j)$ è una funzione di ordinamento (ranking) che associa un numero reale ad ogni coppia costituita da una rappresentazione di una query q_i appartenente a Q e da una rappresentazione di documento dj appartenente a D. Tale ranking definisce un ordine tra i documenti collezionati rispetto alla query q_i.

Il termine indice

- I modelli classici considerano ogni documento come descritto da un insieme di parole chiave rappresentative dette anche termini indice.
- Sia k_i un termine indice, d_j un documento e w_{ij} un peso associato alla coppia (k_i,d_j).
- Questo quantifica l'importanza del termine indice nel descrivere i contenuti semantici del documento: maggiore è il peso maggiormente il termine è significativo ed adatto a descrivere e a rappresentare gli argomenti trattati nel testo.

Peso dei termini (1/2)

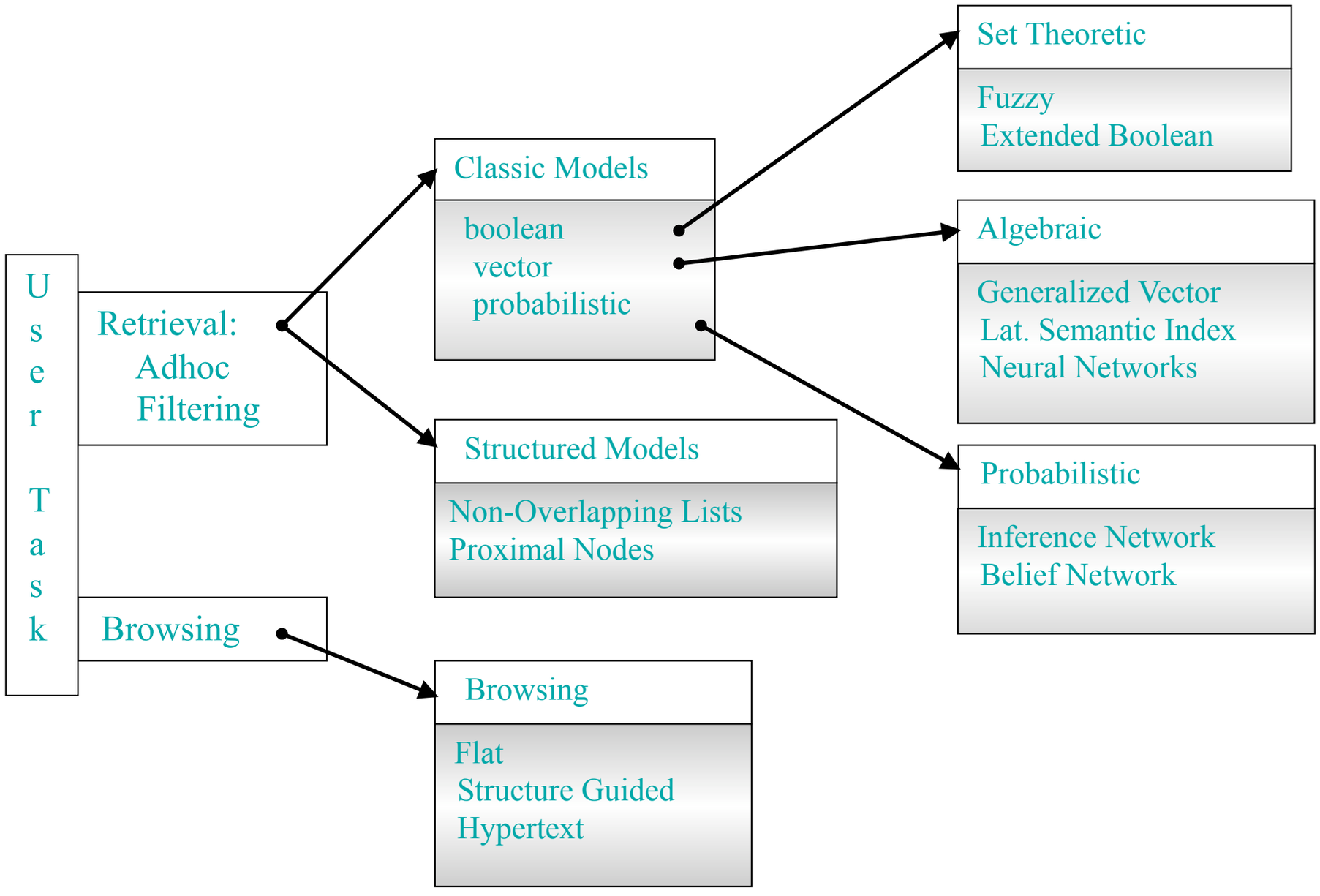
- Sia t il numero di termini indice nel sistema e ki un generico termine indice. $K=\{k_1,...,k_t\}$ è l'insieme di tutti i termini indice. Un peso $w_{ij}>0$ è associato a ogni termine indice k_i di un documento d_j.

- Per un termine indice che non appare nel testo del documento, $w_{ij}=0$.

Peso dei termini (2/2)

- Al documento d_j è associato un vettore di termini indice vec{dj} rappresentato da $vec\{d_j\}=(w_{1j},w_{2j},\ldots,w_{tj})$.
- Sia gi una funzione che restituisce il peso associato al termine indice k_i, in ogni vettore t-dimensionale (cioè $gi(vec\{d_j\})=w_{ij}$)

Modelli IR


User Task
Retrieval: Adhoc Filtering
Browsing
Classic Models
boolean
vector
probabilistic
Structured Models
Non-Overlapping Lists
Proximal Nodes
Browsing
Flat
Structure Guided
Hypertext
Set Theoretic
Fuzzy
Extended Boolean
Algebraic
Generalized Vector
Lat. Semantic Index
Neural Networks
Probabilistic
Inference Network
Belief Network

Modelli di Information Retrieval (1/2)

- Uno dei problemi al centro dei sistemi di IR è quello di predire quali documenti sono rilevanti e quali non lo sono; questa decisione è normalmente dipendente dall'algoritmo di ranking utilizzato, il quale tenta di stabilire, sulla base di una misura di similarità, un semplice ordinamento dei documenti recuperati.
- Tre modelli classici di IR sono:
  - Booleano
  - Vettoriale
  - Probabilistico

Modelli di Information Retrieval (2/2)

- Modello Booleano
  - Nel modello booleano i documenti sono rappresentati come insiemi (set) di parole chiave; pertanto diremo che il modello è di tipo set theoretic.
- Modello Vettoriale
  - Nel modello vettoriale, i documenti sono rappresentati come vettori in uno spazio t-dimensionale; pertanto definiremo questo modello: algebrico.
- Modello Probabilistico
  - Nel modello probabilistico, il fondamento della modellazione della rappresentazione del documento è la teoria della probabilità; diremo pertanto che il modello è probabilistico.

Boolean Retrieval (1/3)

- Il modello booleano è un semplice modello basato sulla teoria degli insiemi e sull'algebra booleana.
- Le query vengono sottomesse come espressioni booleane con precise semantiche, fornendo a questo modello una notevole semplicità e una chiara formalizzazione.

Boolean Retrieval (2/3)

- Per il modello booleano, le variabili dei termini indice sono binarie, cioè w_{ij} appartengono a 0,1.
- Una query q è una convenzionale espressione booleana.
- Sia $\vec{q}_{dnf}$ la forma disgiuntiva normale della query q, $\vec{q}_{cc}$ ognuna delle componenti congiuntive di $\vec{d}_{dnf}$.
- La similarità di un documento d_j alla query è definita come:

$$sim(d_j, q) = \begin{cases} 1 & \text{se } \exists \vec{q_{cc}} | (\vec{q_{cc}} \in \vec{q}_{dnf}) \wedge (\forall k_i : g_i(\vec{d_j}) = g_i(\vec{q}_{cc})) \\ 0 & \text{altrimenti} \end{cases}$$

Boolean retrieval (3/3)

- Vantaggi:
  - Molto pololare (e.g. Search Engines)
  - Semantica facile da comprendere
  - Formalismo semplice e chiaro
  - Facile da implementare

- Svantaggi:
  - Scarsa flessibilità: AND=>Tutti; OR=>qualsiasi
  - Poco espressivo, inadatto ad interrogazioni complesse
  - Nessun controllo sui documenti restituiti (nessun ordinamento)
  - Difficile effettuare un relevance feedback

Esempio:

- Esempio:
  - In quale opera teatrale di Shakespeare ci sono le parole Brutus, Caesar ma non Calpurnia?

BRUTUS AND CAESAR AND NOT CALPURNIA

Matrice di incidenza termine-documento

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
ANTHONY	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 il termine occorre: Calpurnia occorre in Julius Caesar
- 0 altrimenti: Calpurnia non occorre in The Tempest

Vettori di incidenza

- Abbiamo quindi un vettore 0/1 per ogni termine
- per rispondere alla query
  - BRUTUS AND CAESAR AND NOT CALPURNIA

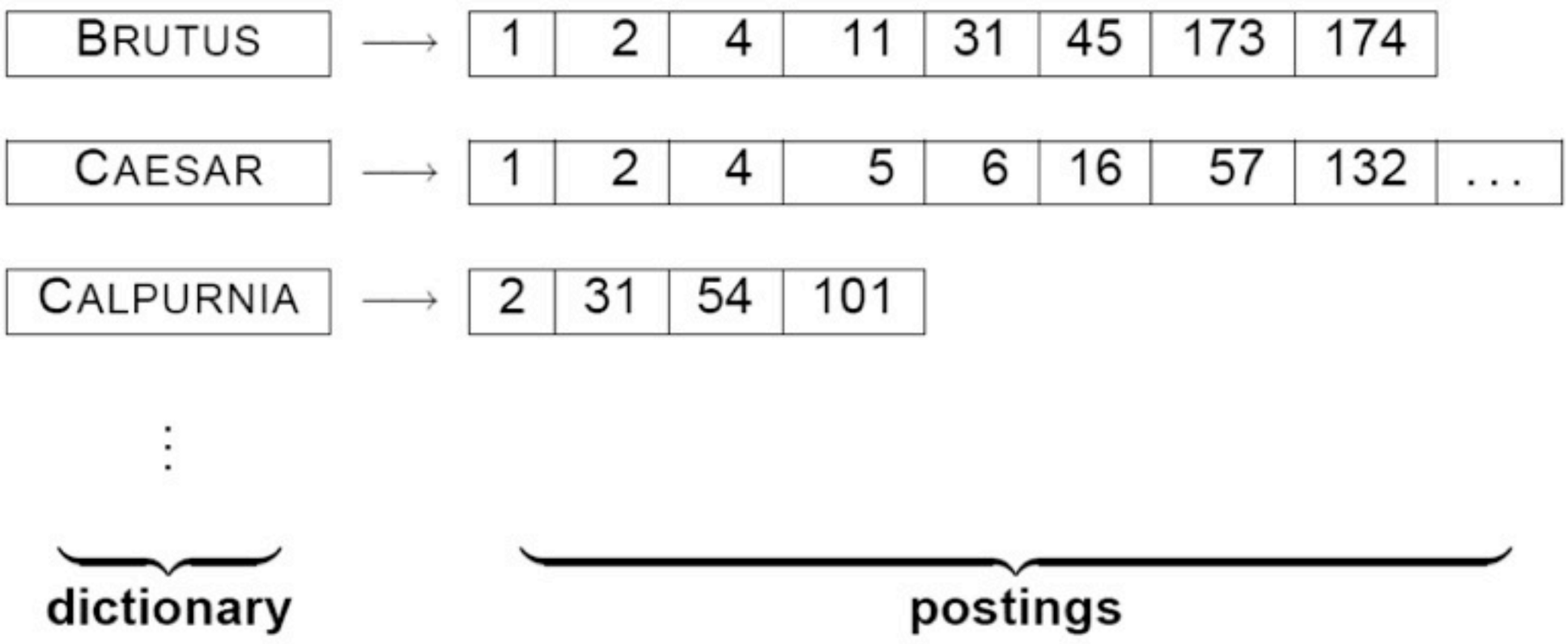
	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	. . .
ANTHONY	1	1	0	0	0	1	
BRUTUS	1	1	0	1	0	0	
CAESAR	1	1	0	1	1	1	
CALPURNIA	0	1	0	0	0	0	
CLEOPATRA	1	0	0	0	0	0	
MERCY	1	0	1	1	1	1	
WORSER	1	0	1	1	1	0	
. . .							

Grandi collezioni di dati

- Consideriamo N=10^6 documenti, ognuno con circa 1000 token
- con una media di 6 bytes per token, includendo spazi e punteggiatura, la dimensione della collezione di documenti raggiunge 6 GB
- supponiamo esistano M=500,000 termini distinti nella collezione
  - la matrice di incidenza termine-documento assume dimensioni non gestibili!! (500,000 x 10^6)

Soluzione: indice inverso

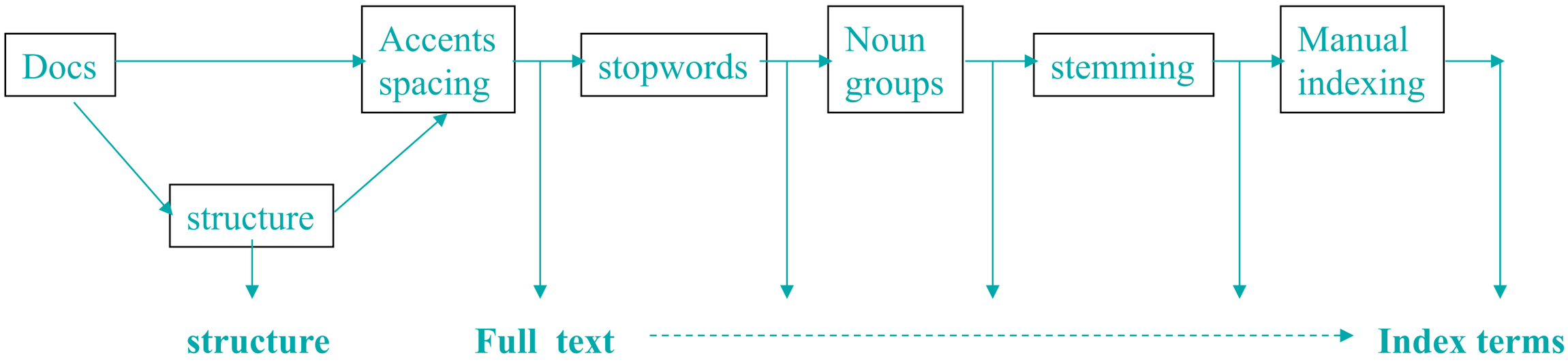
- Per ogni termine t, viene memorizzata una lista di tutti i documenti che contengono t



Costruzione dell'indice inverso

- Colleziona il documento che deve essere indicizzato
  - Friends, Romans, countrymen. So let it be with Caesar.
- Tokenizza il testo
  - Friends Romans countrymen So ...
- Applica un preprocessamento linguistico (stemming), producendo token normalizzati
  - friend roman countryman so ...
- Indicizza i documenti con l'indice inverso

Tokenizzare e preprocessare



- Eliminazione dei caratteri o di simboli di annotazione indesiderati
  - Tag HTML, etichette, punteggiatura etc
- Eliminazione delle parole irrilevanti (stopwords)
  - Esempio: un, il, esso, essa.
- Rilevamento delle frasi comuni (Noun groups)
- Stemming dei token in “radici”
  - **Computational → Compute**

Esempio: tokenizzazione e preprocessamento

Doc 1. I did enact Julius Caesar: I was killed i' the Capitol; Brutus killed me.
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
Doc 2. so let it be with caesar the noble brutus hath told you caesar was ambitious

Ordina postings
e crea la lista delle frequenze

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

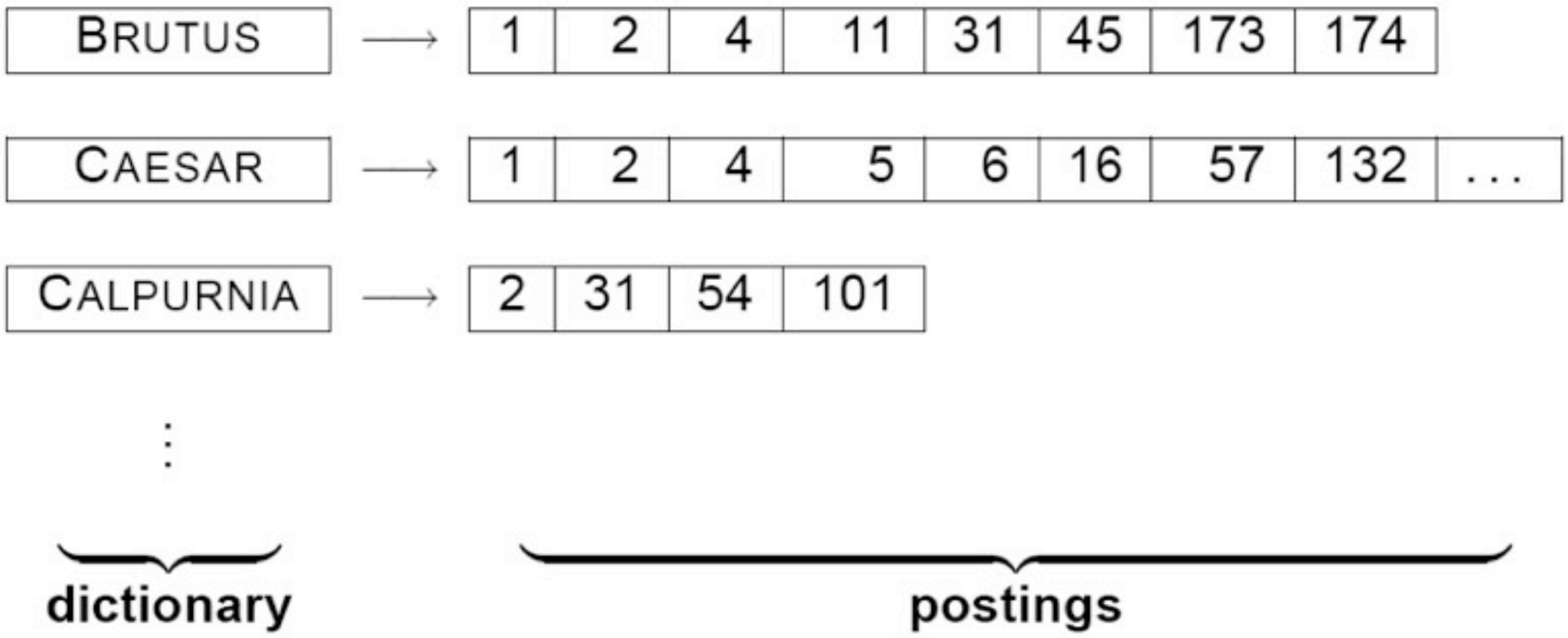
⟹

term	docID
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	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
did	1	→	1
enact	1	→	1
hath	1	→	2
I	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

Inserire il risultato nel dizionario e nel postings file



Esercizio 1

- Scrivere l'indice inverso utilizzando i seguenti documenti:

- **Doc1**: new home sales top forecasts
- **Doc2**: home sales rise in july
- **Doc3**: increase in home sales in july
- **Doc4**: july new home sales rise

Soluzione

- forecast -> 1
- home -> 1 -> 2 -> 3 -> 4
- in -> 2 -> 3
- increase -> 3
- july -> 2 -> 3
- new -> 1 -> 4
- rise -> 2 -> 4
- sale -> 1 -> 2 -> 3 -> 4
- top -> 1

Esercizio 2

- Scrivere la matrice termine-documento e l'indice inverso per i seguenti documenti:

- **Doc1**: breakthrough drug for schizophrenia
- **Doc2**: new schizophrenia drug
- **Doc3**: new approach for treatment of schizophrenia
- **Doc4**: new hopes for schizophrenia patients

Soluzione

	d1	d2	d3	d4
Approach	0	0	1	0
breakthrough	1	0	0	0
drug	1	1	0	0
for	1	0	1	1
hopes	0	0	0	1
new	0	1	1	1
of	0	0	1	0
patients	0	0	0	1
schizophrenia	1	1	1	1
treatment	0	0	1	0

Approach	3			
breakthrough	1			
drug	1	2		
for	1	3	4	
hopes	4			
new	2	3	4	
of	3			
patients	4			
schizophrenia	1	2	3	4
treatment	3			

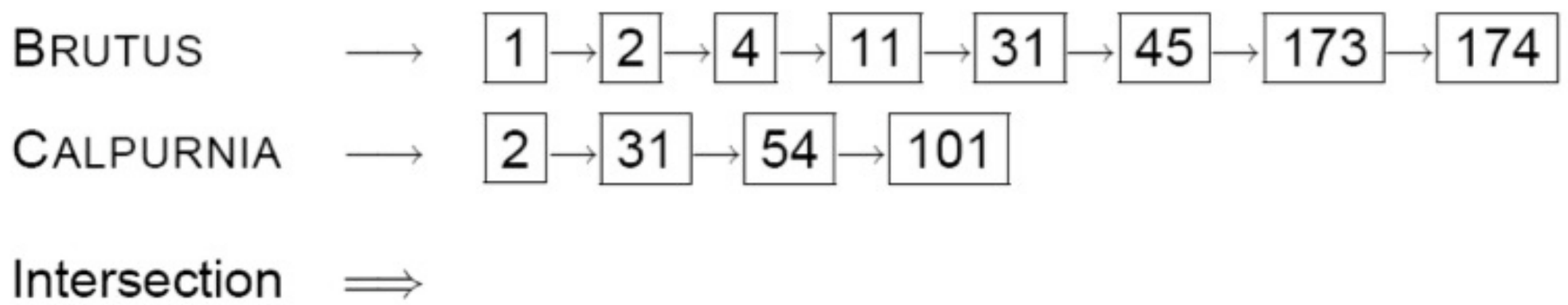
Query con termini in forma congiuntiva

- Consideriamo la query: BRUTUS AND CALPURNIA
- Passi necessari per localizzare i documenti che rispondono alla query:

1. localizza BRUTUS nel dizionario
2. ricerca la relativa lista dei posting
3. localizza CALPURNIA nel dizionario
4. ricerca la relativa lista dei posting
5. interseca le due liste
6. restituisci l'intersezione all'utente

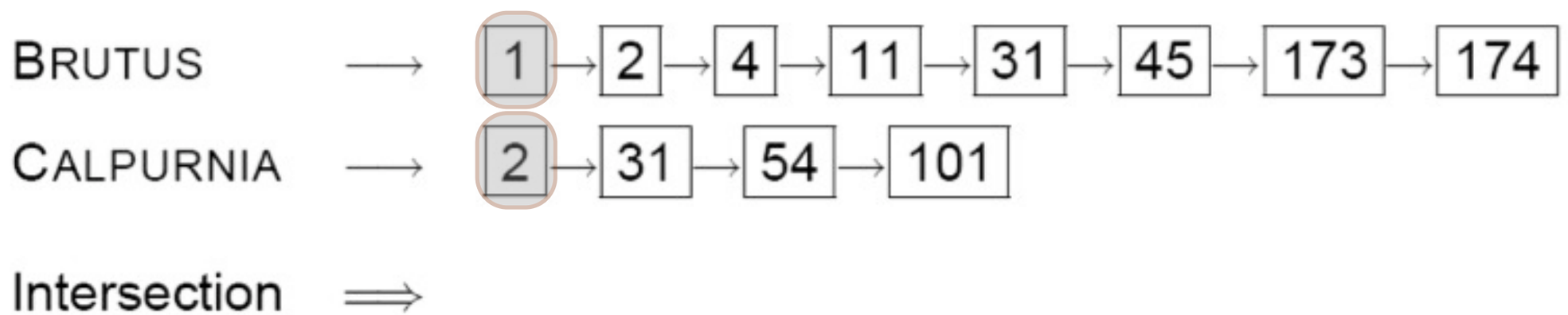
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



Algoritmo intersect

- Query: BRUTUS AND CALPURNIA

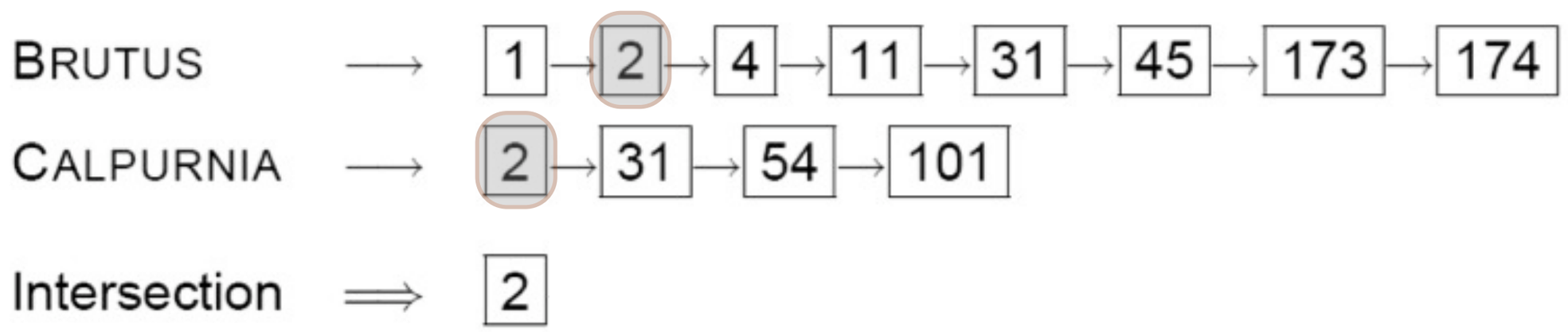


Algoritmo intersect

- Query: BRUTUS AND CALPURNIA

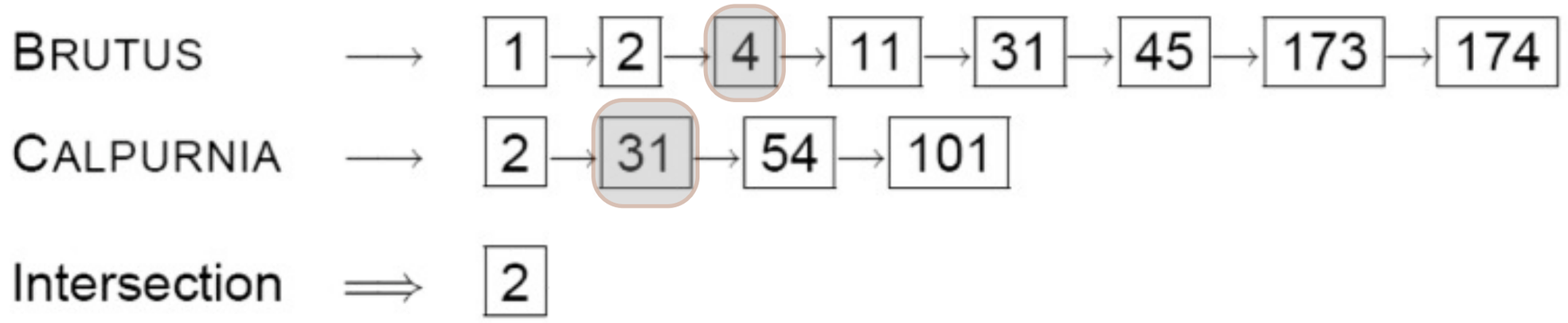
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



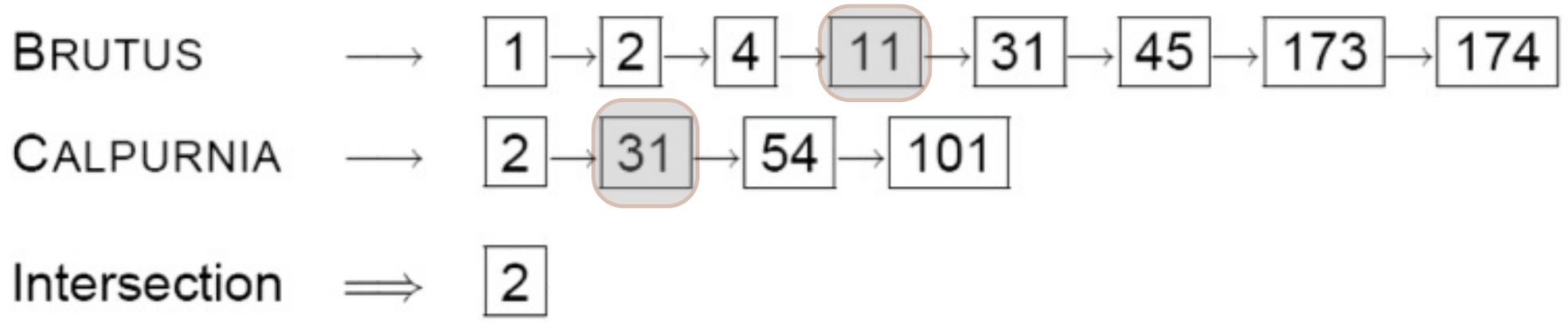
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



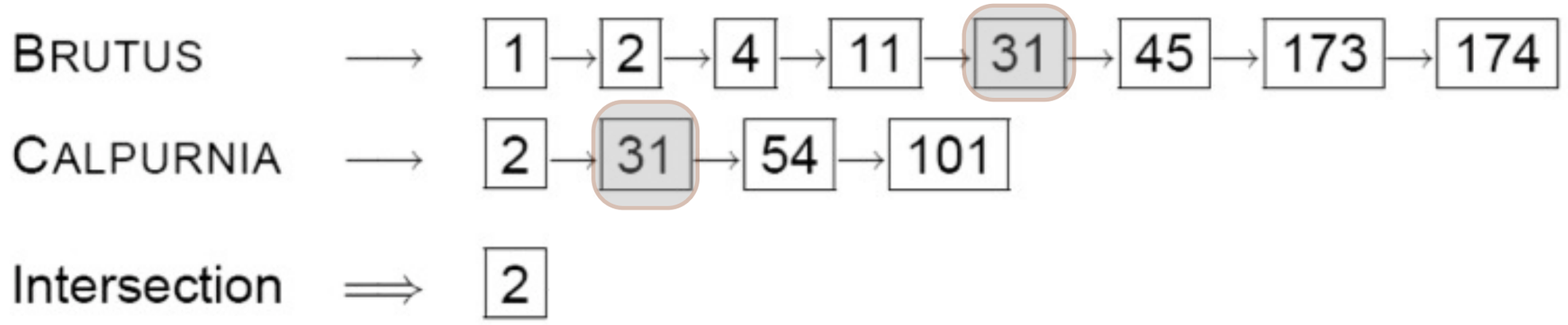
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



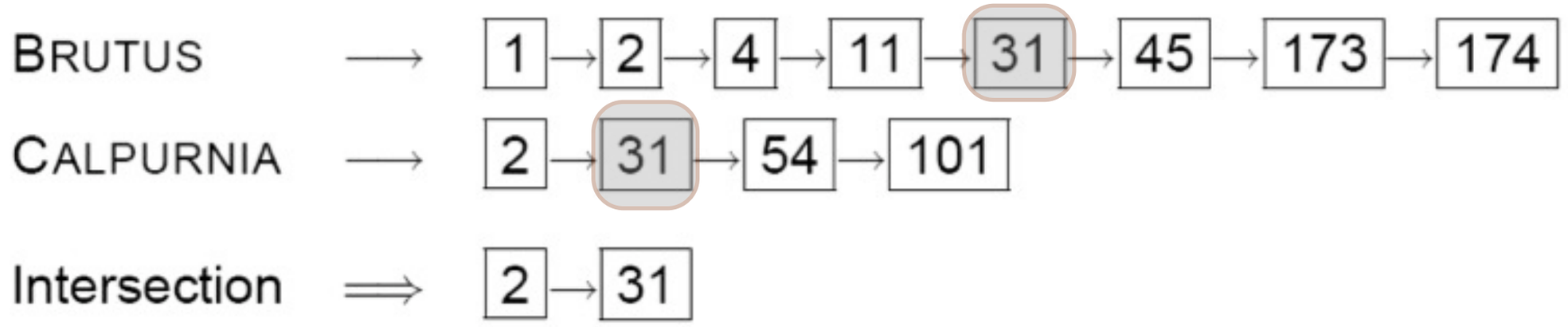
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



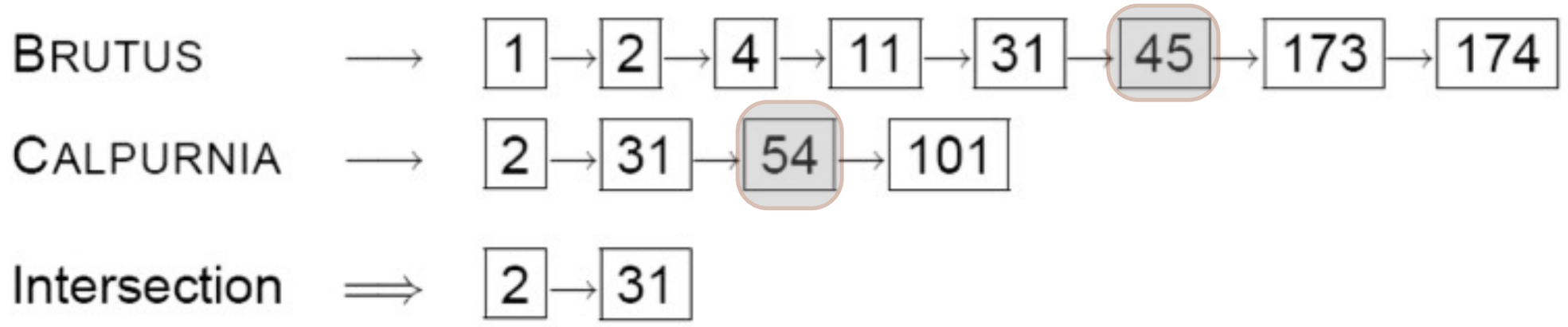
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



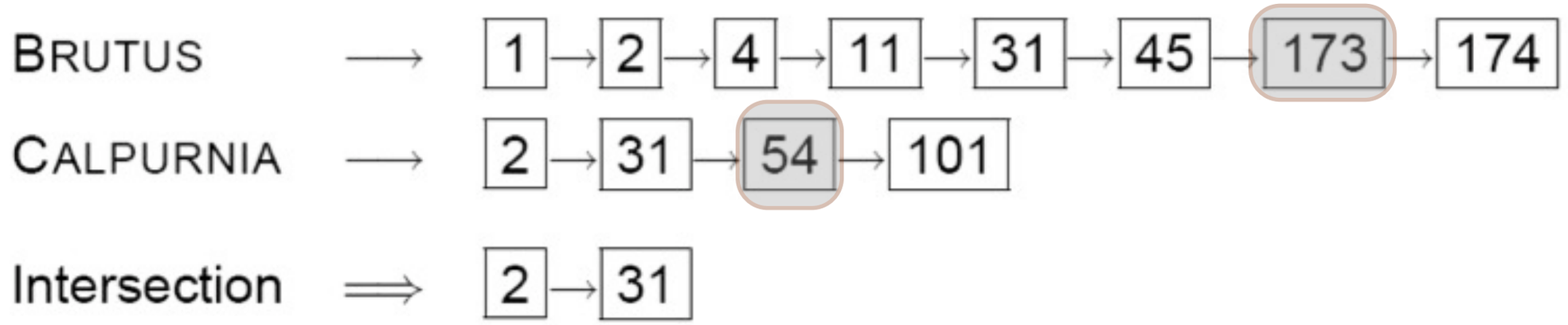
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



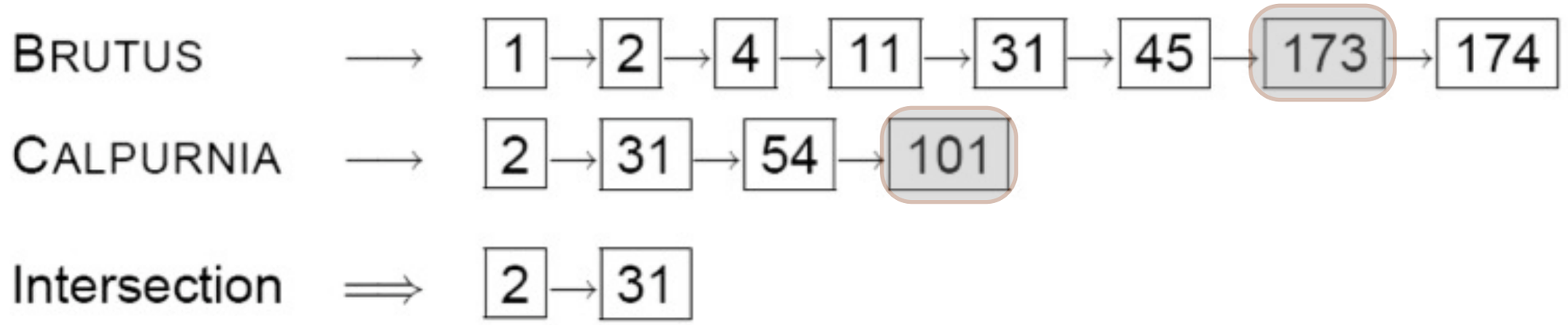
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



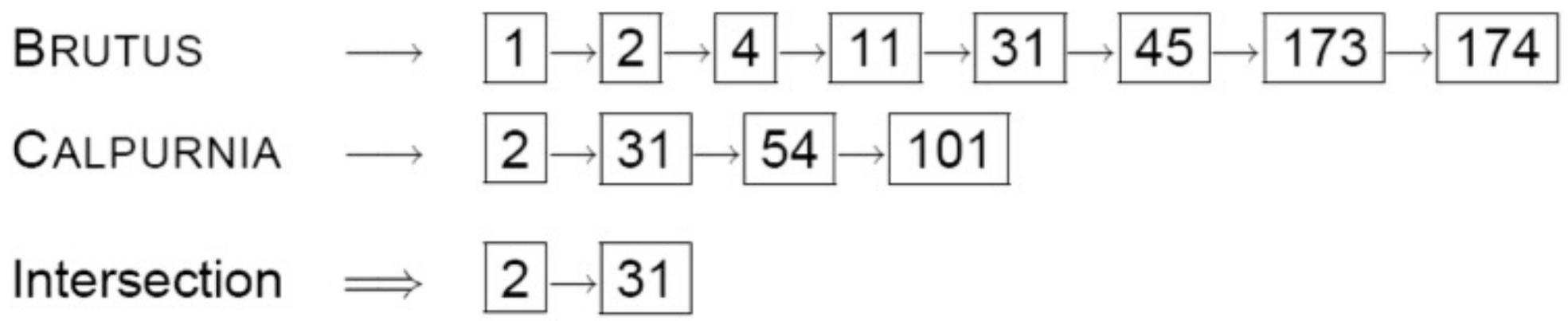
Algoritmo intersect

- Query: BRUTUS AND CALPURNIA



Algoritmo intersect

- Algoritmo lineare nella lunghezza delle liste dei posting
- Le liste devono essere ordinate!

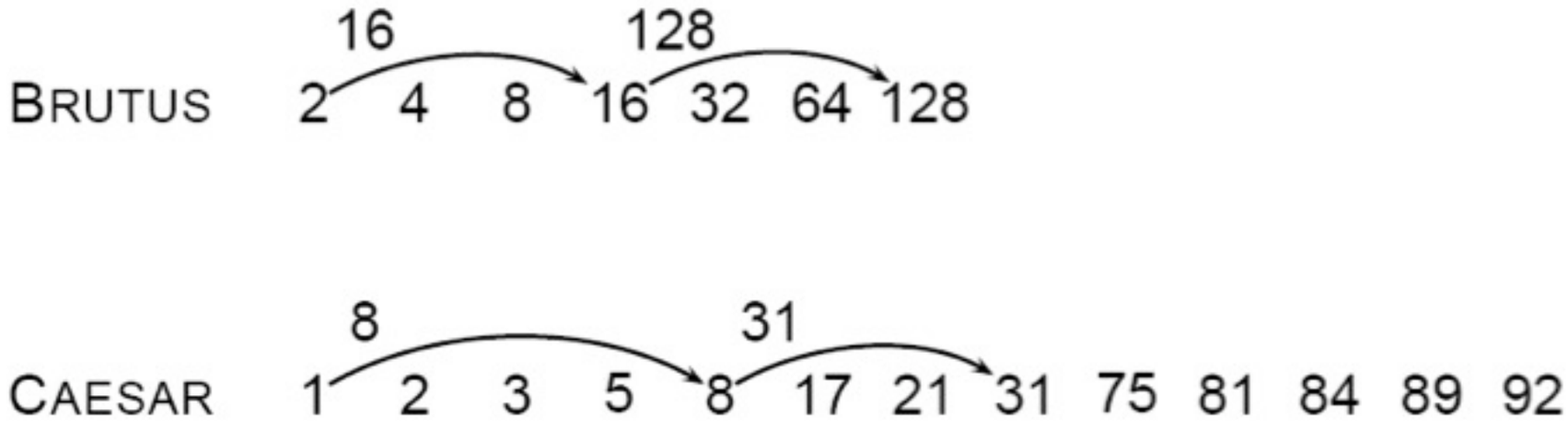


Pseudocodifica intersezione di due liste di posting

```
INTERSECT(p1, p2)
 1  answer ← ⟨ ⟩
 2  while p1 ≠ NIL and p2 ≠ NIL
 3  do if docID(p1) = docID(p2)
 4        then ADD(answer, docID(p1))
 5             p1 ← next(p1)
 6             p2 ← next(p2)
 7        else if docID(p1) < docID(p2)
 8                then p1 ← next(p1)
 9                else p2 ← next(p2)
10  return answer
```

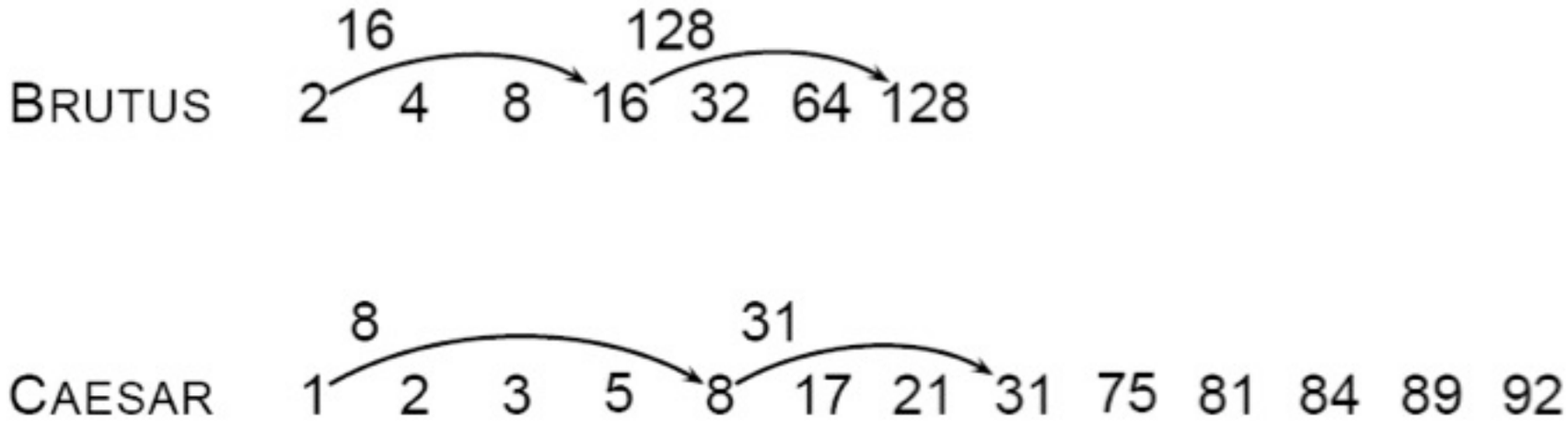
Skip List

- Intuizione:



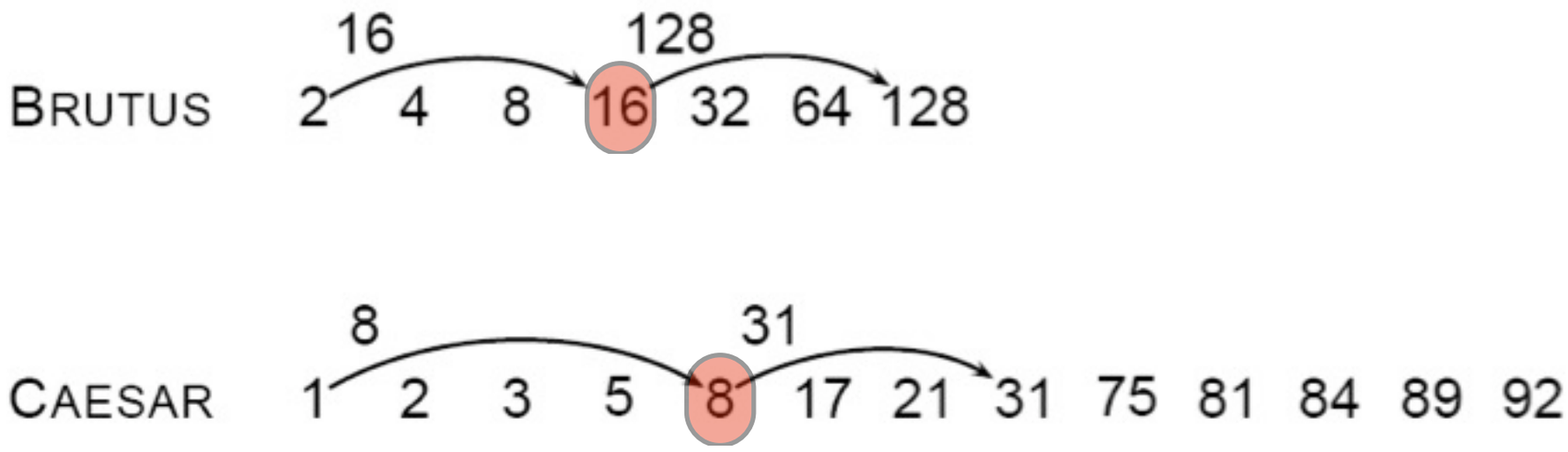
Skip List

- Intuizione:



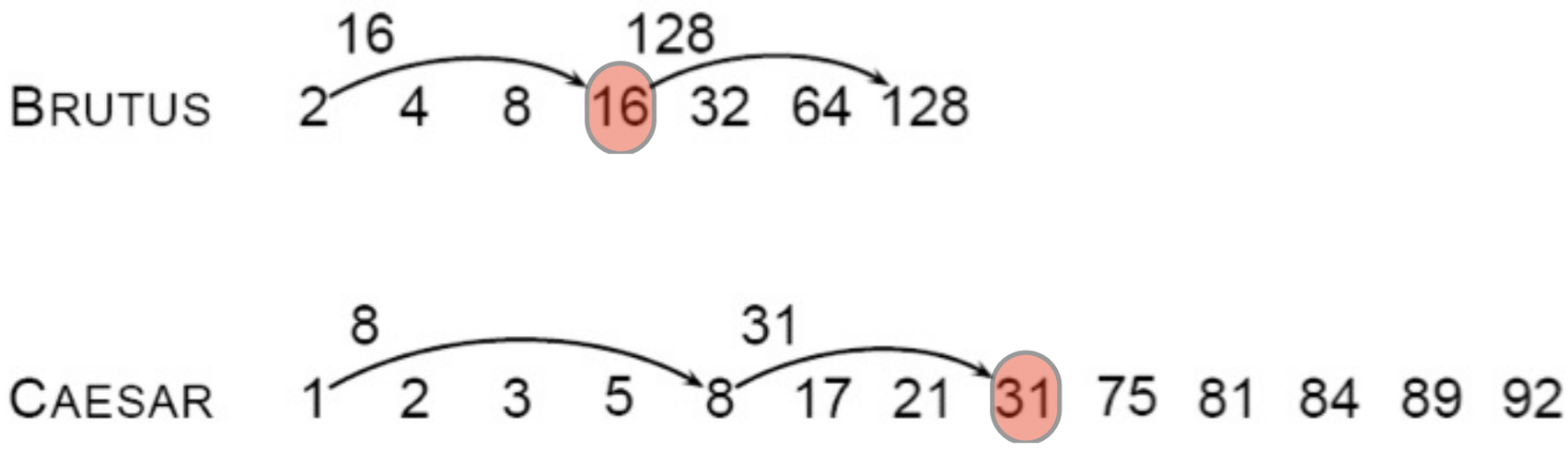
Skip List

- Intuizione:



Skip List

- Intuizione:



Esercizio

- Scrivere la pseudo codifica dell'algoritmo di intersezione con skip list
- utilizzare la funzione:
  - boolean hasSkip(p1)
  - che identifica se nel nodo p1 è presente una skip list.

Esercizio

- Scrivere l'algoritmo di unione di liste di posting
  - UNION(x,y)

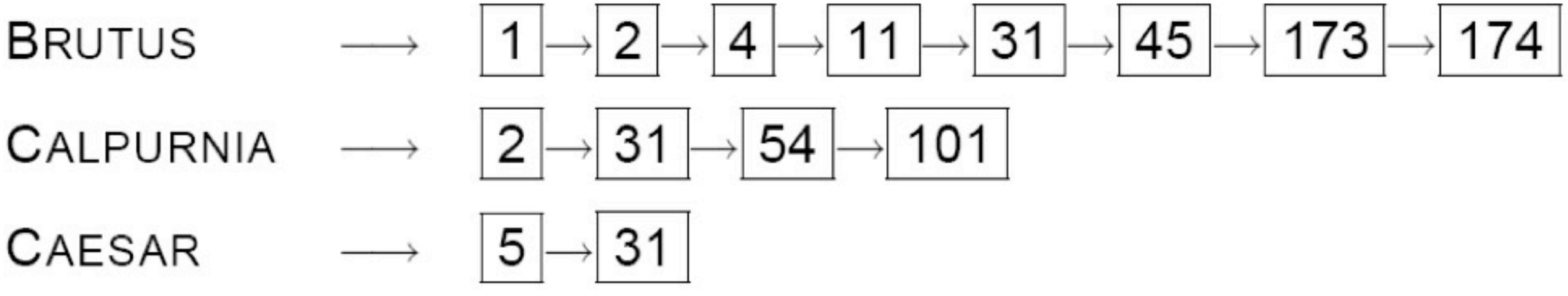
```
1answer<- ( )
2while x!=NIL and y!=NIL
3do if docID(x)=docID(y)
4 then ADD(answer,docID(x))
5 x<- next(x)
6 y<-next(y)
7 else if docID(x)<docID(y)
8 then ADD(answer,docID(x))
9 x<- next(x)
10 else ADD(answer,docID(y))
11 y<-next(y)
12 return(answer)
```

Ottimizzazione della query

- Qual è il miglior modo di processare una query?
- Consideriamo una query composta dalla congiunzione di n termini (n>2)
- Per ogni termine otteniamo la posting list ed esguiamo l'algoritmo di intersezione.
- Consideriamo come esempio la seguente query:
  - BRUTUS **AND** CALPURNIA **AND** CAESAR

Ottimizzazione della query

- BRUTUS **AND** CALPURNIA **AND** CAESAR
- Processo le coppie di liste in ordine decrescente di frequenza dei relativi termini
  - quindi seguirò il seguente ordine:
    - CAESAR, CALPURNIA, BRUTUS



Pseudocodifica congiunzione di n termini

```
INTERSECT(⟨t_1, ..., t_n⟩)
1  terms ← SORTBYINCREASINGFREQUENCY(⟨t_1, ..., t_n⟩)
2  result ← postings(first(terms))
3  terms ← rest(terms)
4  while terms ≠ NIL and result ≠ NIL
5  do result ← INTERSECT(result, postings(first(terms)))
6       terms ← rest(terms)
7  return result
```

Tolleranza agli errori nelle query

- In una query (o in un documento) possono essere contenuti degli errori di battitura
- Definisci la similarità tra parole o stringhe arbitrarie mediante:
  - Distanza di Editing
  - Sottosequenza comune più lunga (Longest Common Subsequence, LCS)
  - Supportano in genere la ricerca per prossimità con un limite sulla similarità tra stringhe.

Spelling correction

- Il confronto tra sequenze è fondamentale nel task di spelling correction
- Possibili obiettivi:
  - misurare la “similarità” tra le sequenze
    - allineamento
  - misurare la “diversità” tra le sequenze
    - distanza di edit
  - trovare parti comuni alle sequenze
    - pattern discovery
    - allineamento locale

Edit (Levenshtein) Distance

- E’ il minimo numero di caratteri di cancellazione, aggiunta o rimpiazzamento che sono necessari a rendere due stringhe identiche
  - “misspell” VS “mispell” distanza = 1
  - “misspell” VS “mistell” distanza = 2
  - “misspell” VS “misspelling” distanza = 3
- Approccio efficiente: usa una tecnica di programmazione dinamica. La complessità temporale è O(mn) dove m ed n sono le lunghezze delle due stringhe in esame.

Levenshtein
calcolo matrice

- Calcolo matrice di levenstein per le parole
  - cats
  - fast

		f	a	s	t
	0	1	2	3	4
c	1	1	2	3	4
a	2	2	1	2	3
t	3	3	2	2	2
s	4	4	3	2	3

Pseudocodice distanza di levenshtein

```
LevenshteinDistance(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1]}
 9            else m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1] + 1}
10  return m[|s1|, |s2|]
```

Operations: insert, delete, replace, copy

Pseudocodice distanza di levenshtein

```
LevenshteinDistance(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1]}
 9            else m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1] + 1}
10  return m[|s1|, |s2|]
```

Operations: insert, delete, replace, copy

Pseudocodice
distanza di levenshtein

```
LEVENSHTEINDISTANCE(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1]}
 9            else m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1] + 1}
10  return m[|s1|, |s2|]
```

Operations: insert, delete, replace, copy

Pseudocodice
distanza di levenshtein

```
LEVENSHTEINDISTANCE(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1]}
 9            else m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1] + 1}
10  return m[|s1|, |s2|]
```

Operations: insert, delete, replace, copy

Pseudocodice distanza di levenshtein

```
LEVENSHTEINDISTANCE(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1]}
 9            else m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1] + 1}
10  return m[|s1|, |s2|]
```

Operations: insert, delete, replace, copy

Esecuzione algoritmo

		f	a	s	t
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
c	**1** / **1**	*1* *2* / *2* *1*	**2** 3 / **2** **2**	**3** 4 / **3** **3**	**4** 5 / **4** **4**
a	**2** / **2**	**2** **2** / 3 **2**	*1* *3* / *3* *1*	*3* *4* / *2* *2*	4 5 / **3** **3**
t	**3** / **3**	**3** **3** / 4 **3**	3 **2** / 4 **2**	*2* *3* / *3* *2*	*2* *4* / *3* *2*
s	**4** / **4**	**4** **4** / 5 **4**	4 **3** / 5 **3**	**2** 3 / 4 **2**	*3* *3* / *3* *3*

cost of getting here from my upper left neighbor (copy or replace)	cost of getting here from my upper neighbor (delete)
cost of getting here from my left neighbor (insert)	the minimum of the three possible “movements”; the cheapest way of getting here

Esercizio

- Definire la matrice di levenshtein per le parole
  - paris
  - alice

Soluzione

		a	l	i	c	e
	0	1 1	2 2	3 3	4 4	5 5
p	1 1	1 2 2 1	2 3 2 2	3 4 3 3	4 5 4 4	5 6 5 5
a	2 2	1 2 3 1	2 3 2 2	3 4 3 3	4 5 4 4	5 6 5 5
r	3 3	3 2 4 2	2 3 3 2	3 4 3 3	4 5 4 4	5 6 5 5
i	4 4	4 3 5 3	3 3 4 3	2 4 4 2	4 5 3 3	5 6 4 4
s	5 5	5 4 6 4	4 4 5 4	4 3 5 3	3 4 4 3	4 5 4 4

Longest Common Subsequence (LCS)

- Lunghezza della più lunga sottosequenza di caratteri condivisa dalle due stringhe
- Una sottosequenza si ottiene da una stringa cancellando 0 o più caratteri
- esempi:
  - “misspell” VS “mispell” 7
  - “misspelled” VS “misinterpretted” 7
  - poiché la LCS è “mis...p...e...ed”

Vector-Space Model

- Definisci tutti i termini presenti nella base di documenti. Vocabolario (V)
- Assegna i termini ai vettori ortogonali dello spazio
  - Dimensioni = |V| = N
- Ogni termine i-esimo riceve un peso in un documento j-esimo (o query), w_{ij}
- I documento o le query sono vettori N-dimensionali

$$d_j = (w_{1_j}, w_{2_j}, \ldots, w_{N_j})$$

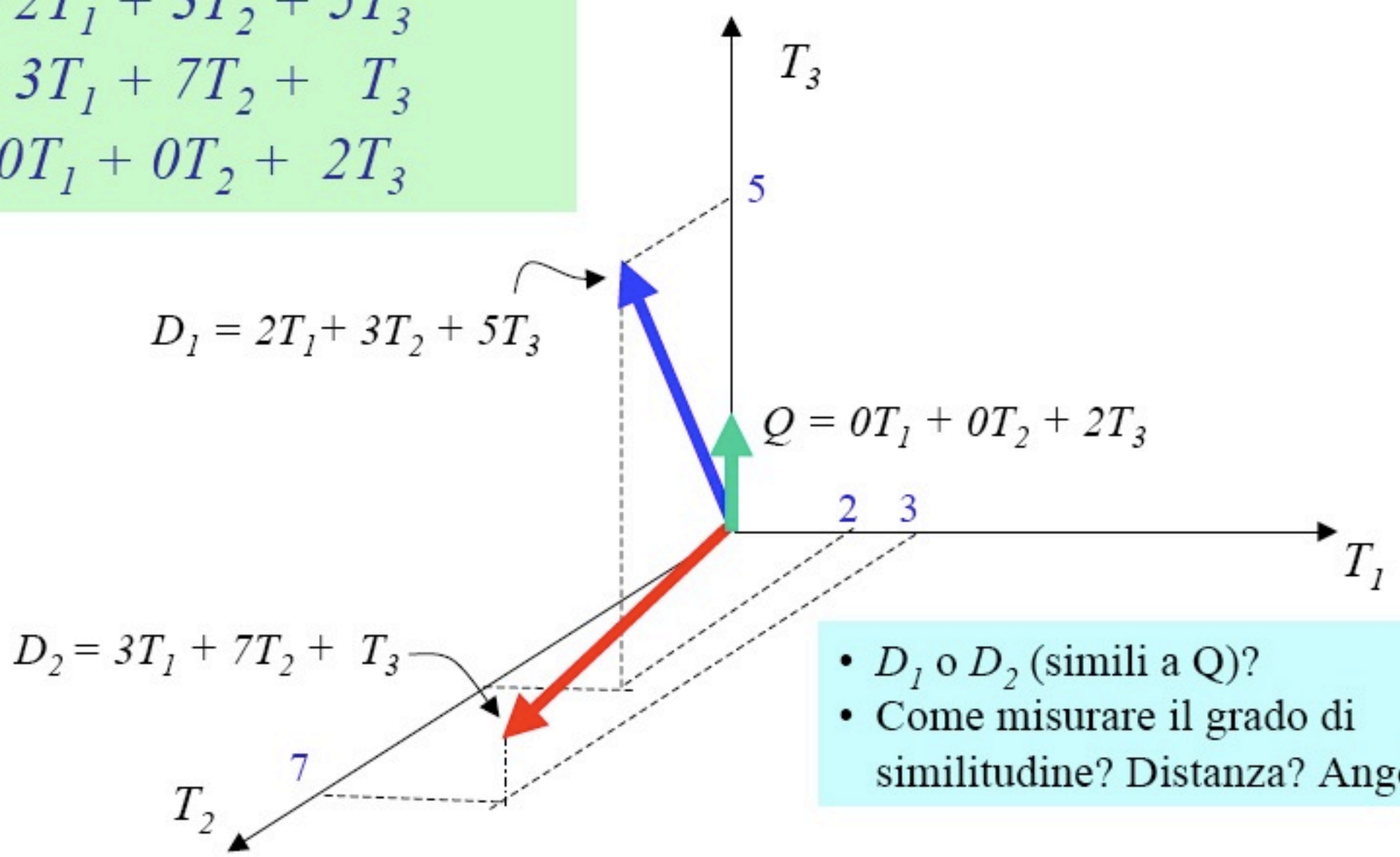
Interpretazione geometrica

Esempio:

$D_1 = 2T_1 + 3T_2 + 5T_3$

$D_2 = 3T_1 + 7T_2 + T_3$

$Q = 0T_1 + 0T_2 + 2T_3$



- D_1 o D_2 (simili a Q)?
- Come misurare il grado di similitudine? Distanza? Angolo?

La collezione dei documenti esempio caso binario

- Una collezione di n documenti viene rappresentata nel VSM dalla matrice termine-documento

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	. . .
ANTHONY	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	
. . .							

La collezione dei documenti esempio caso ‘frequenze’

- Il peso da associare al termine può essere calcolato con formule diverse (nell’esempio il caso delle frequenze!).

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	. . .
ANTHONY	157	73	0	0	0	1	
BRUTUS	4	157	0	2	0	0	
CAESAR	232	227	0	2	1	0	
CALPURNIA	0	10	0	0	0	0	
CLEOPATRA	57	0	0	0	0	0	
MERCY	2	0	3	8	5	8	
WORSER	2	0	1	1	1	5	
. . .							

Vector space model

- Un modello specifico di VSM si distingue da altri relativamente alle scelte di:
  - Pesatura dei termini nelle interrogazioni
  - Pesatura dei termini nei documenti
  - Funzione di similarità (metrica)
  - Criterio di accettazione (i.e. soglia di rilevanza)

Come pesare i termini? Frequenza

- Più i termini sono in un documento, più importanti essi tendono ad essere, i.e. più significativi per il contenuto

$$f_{ij} = \text{frequenza del termine i nel documento j}$$

- Per normalizzare la frequenza (tf), term frequency, attraverso il corpus:

$$tf_{ij} = \frac{f_{ij}}{\max_k\{f_{kj}\}}$$

Come pesare i termini? Inverse Document Frequency

- I termini che appaiono in molti documenti differenti sono meno indicativi del contenuto (e.g. a, da, io, per, questo, etc...)

- df_i: document frequency del termine i
  - numero di documenti che contengono il termine i
- idf_i: inverse document frequency del termine i,
  - $\log_2(N/df_i)$
  - (N: numero totale di documenti)

Inverse document frequency

- L'inverse document frequency (idf_i) è un indice del potere di discriminazione di un termine
  - Se $df_i = N$ allora segue che $idf_i=0$
- Il logaritmo contiene l'effetto della frequenza

Pesatura tf-idf

- Un indicatore che cattura le precedenti proprietà è il fattore tf-idf:
  - $w_{ij} = tf_{ij} \times idf_i = tf_{ij} \times \log_2(N/df_i)$
- Un termine che occorre frequentemente in un documento ma raramente nell'intera collezione riceve un peso alto

Esercizio

- Un documento ha i seguenti termini con le seguenti frequenze:
  - A(3), B(2), C(1)
- Assumiamo una collezione di 10000 docs in cui le frequenze globali sono
  - A(50), B(1300), C(250)
- Calcolare per A, B e C i valori di tf, idf e tf-idf

Soluzione

- A: tf=3/3; idf=log(10000/50)=5.3; tf-idf=5.3
- B: tf=2/3; idf=log(10000/1300)=2.0; tf-idf=1.3
- C: tf=1/3; idf=log(10000/250)=3.7; tf-idf=1.2

Il vettore della query

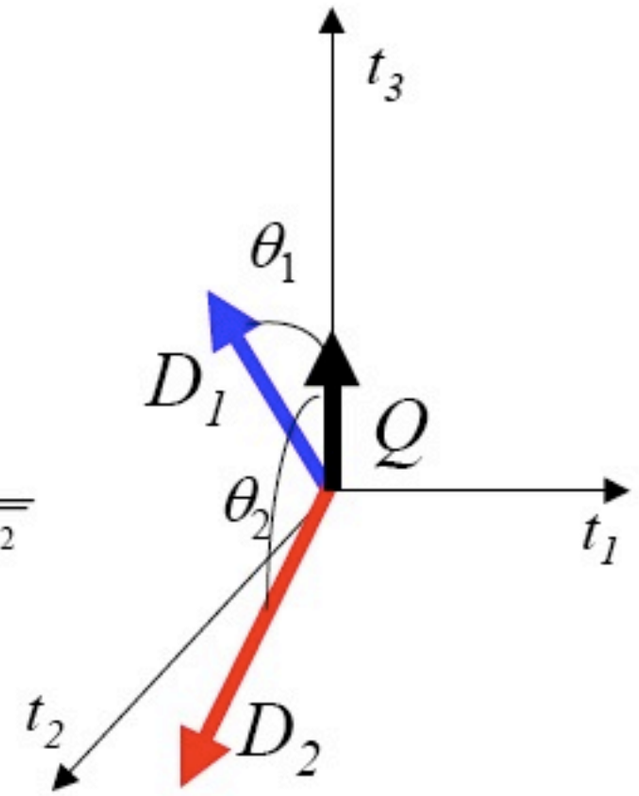
- Il vettore che rappresenta la query è tipicamente trattato come un documento e pesato come un documento e pesato tramite il fattore tf-idf.
- Alternativa: fornire i pesi assengati dagli utenti ai termini della query.

Metrica di similarità

- Una metrica di similarità è una funzione che calcola il grado di similitudine tra due vettori
- Grazie all'uso di una metrica di similarità tra la query ed ogni documento è possibile:
  - ordinare i documenti dal più simile al meno simile
  - impostare una soglia al di sopra della quale respingere i documenti (per es. per controllare la numerosità dei risultati).

Coseno similarità

- Misura il coseno dell'angolo tra due vettori.



$$\text{CosSim}(\boldsymbol{d}_j, \boldsymbol{q}) = \frac{\vec{d} \times \vec{q}}{|\vec{d}| \cdot |\vec{q}|} = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \cdot \sum_{i=1}^{t} w_{iq}^2}}$$

$D_1 = 2T_1 + 3T_2 + 5T_3$ $\quad \text{CosSim}(D_1, Q) = 10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$

$D_2 = 3T_1 + 7T_2 + 1T_3$ $\quad \text{CosSim}(D_2, Q) = 2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$

$Q = 0T_1 + 0T_2 + 2T_3$

D_1 e' 6 volte migliore di D_2 secondo la cosine similarity

Implementazione

- Converti tutti i documenti della collezione D in vettori d_j pesati tramite tf-idf, per le keyword nel vocabolario V.
- Converti l'interrogazione (query) in un vettore q pesato tramite tf-idf.
- Per ogni $d_j \in D$:
  - Calcola s_j=CosSim(d_j,q)
- Ordina i documenti d_j in ordine decrescente secondo s_j e mostra i primi documenti ottenuti all'utente.
- Complessità: O(|V||D|)

Pregi VSM

- Approccio semplice ma formalmente ben definito
- Tiene conto sia delle frequenze lessicali locali sia di quelle globali
- Supporta matching parziali e risultati ordinati
- L'accuratezza empirica è molto buona
- Consente una implementazione efficiente anche per grandi collezioni di documenti

Difetti VSM

- Manca il supporto per le informazioni di tipo semantico
- Non è sensibile all'informazione sintattica
- Manca il supporto definito dal modello booleano
  - "un termine DEVE comparire all'interno del documento"

Retrieval statistico

- Basato sulla similarità tra query e documento
- I documenti trovati sono ordinati in base alla similarità rispetto alla query
- La similarità è basata sulla frequenza di occorrenze delle keywords nella query e nel documento
- Relevance Feedback:
  - Aggiungi alla query i documenti più rilevanti.
  - Rimuovi gli irrilevanti dalla query

Modelli Statistici

- Un documento è rappresentato come bag of words (multi) insiemi di parole (con frequenze).
- Bag → occorrenze multiple
- L'utente inserisce i termini desiderati con un peso opzionale
  - Termini pesati dell'intorregazione
    - Q: <database 0.5, testo 0.8, informazione 0.2>
  - Termini non pesati
    - Q: <database, testo, informazione>
  - Nessuna condizione booleana alla query

Modello probabilistico

- Data una query, vogliamo trovare i documenti che hanno la maggior probabilità di essere rilevanti.
- Vogliamo quindi calcolare: $P(R = true|D, Q)$
- In cui D è un documento, Q una query e R una variabile casuale booleana che indica la rilevanza.
- Fatto questo sarà possibile presentare i risultati sotto forma di lista ordinata sulla probabilità di rilevanza.

Modellazione del linguaggio (1/3)

- Usando r per denotare il valore R=true, possiamo riscrivere la probabilità come segue:

$$P(r|D,Q) = \frac{P(D,Q|r)P(r)}{P(D,Q)}$$

- per la regola di Bayes

$$= \frac{P(Q|D,r)P(D|r)P(r)}{P(D,Q)}$$

- per la regola della catena

$$= \alpha \frac{P(Q|D,r)P(r|D)}{P(D,Q)}$$

- per la regola di Bayes, per D fissati

Modellazione del linguaggio (2/3)

- Massimizzare P(r|D,Q) è equivalente a massimizzare il rapporto P(r|D,Q)/P(¬r|D,Q).
- Questo significa che possiamo ordinare i documenti in base al punteggio:

$$\frac{P(r|D,Q)}{P(\neg r|D,Q)} = \frac{P(Q|D,r)P(r|D)}{P(Q|D,\neg r)P(\neg r|D)}$$

Modellazione del linguaggio (3/3)

- Con l'ipotesi che i documenti irrilevanti siano indipendenti dalla query avremo:
  - $P(D,Q|\neg r)=P(D|\neg r)P(Q|\neg r)$

$$\frac{P(r|D,Q)}{P(\neg r|D,Q)} = P(Q|D,r) \times \frac{P(r|D)}{P(\neg r|D)}$$

- Il primo fattore è la probabilità indipendente dalla query che il documento sia rilevante
- Il secondo fattore è la probabilità indipendente dalla query che il documento sia rilevante!

Calcolo della rilevanza

- Per calcolare la probabilità di una query dato un documento rilevante è sufficiente moltiplicare la probabilità delle parole nella query:

$$P(Q|D,r) = \prod_j P(Q_j|D,r)$$

$$\frac{P(r|D,Q)}{P(\neg r|D,Q)} = \prod_j P(Q_j|D,r) \times \frac{P(r|D)}{P(\neg|D)}$$

Confronto tra i modelli classici

- Il modello booleano non sostiene matching parziali e sembra il modello più debole
- Salton e Buckley dimostrano sperimentalmente che il VSM è più performante rispetto al modello probabilistico.
- Il Vector Space Model (e sue varianti) sembrano il modello empiricamente più accurato.

Alcune domande aperte

- Come determinare le parole importanti all'interno di un documento?
  - Sensi VS Parole
  - Sequenze (n-grammi di parole, espressioni idiomatiche) -> termini
- Rilevanza dei termini in un documento VS rilevanza dei termini nell'intera collezione
- Similitudine tra interrogazioni e documenti

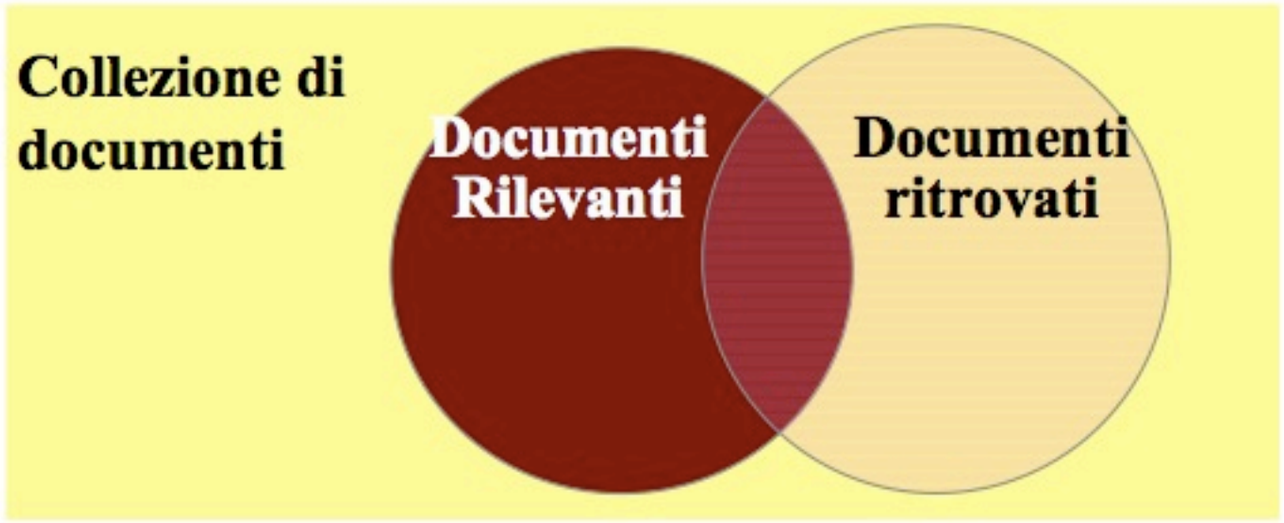
Performance evaluation motivazioni

- Tra i diversi sistemi/modelli/algoritmi di IR quali sono i migliori?
- La migliore componente per:
  - la funzione di ranking
  - selezione dei termini
  - pesatura dei termini
- Quando interrompere la lettura della lista ordinata di documenti ritrovati?

Problemi nella valutazione dei sistemi di IR

- Efficacia dipende dalla rilevanza degli elementi ritrovati
- La rilevanza appare come una funzione continua e non binaria
- Specialmente se intesa in modo categoriale la rilevanza è molto difficile da modellare
- La rilevanza è:
  - soggettiva: dipende dal punto di vista dell'utente
  - contestuale: dipende dai requisiti dell'utente
  - cognitiva: viene percepita a subita dall'utente
  - dinamica: varia nel tempo

Precision - Recall



	trovati	non trovati
irrilevanti	trovati & irrilevanti	Non trovati & irrilevanti
rilevanti	trovati & rilevanti	non trovati & rilevanti

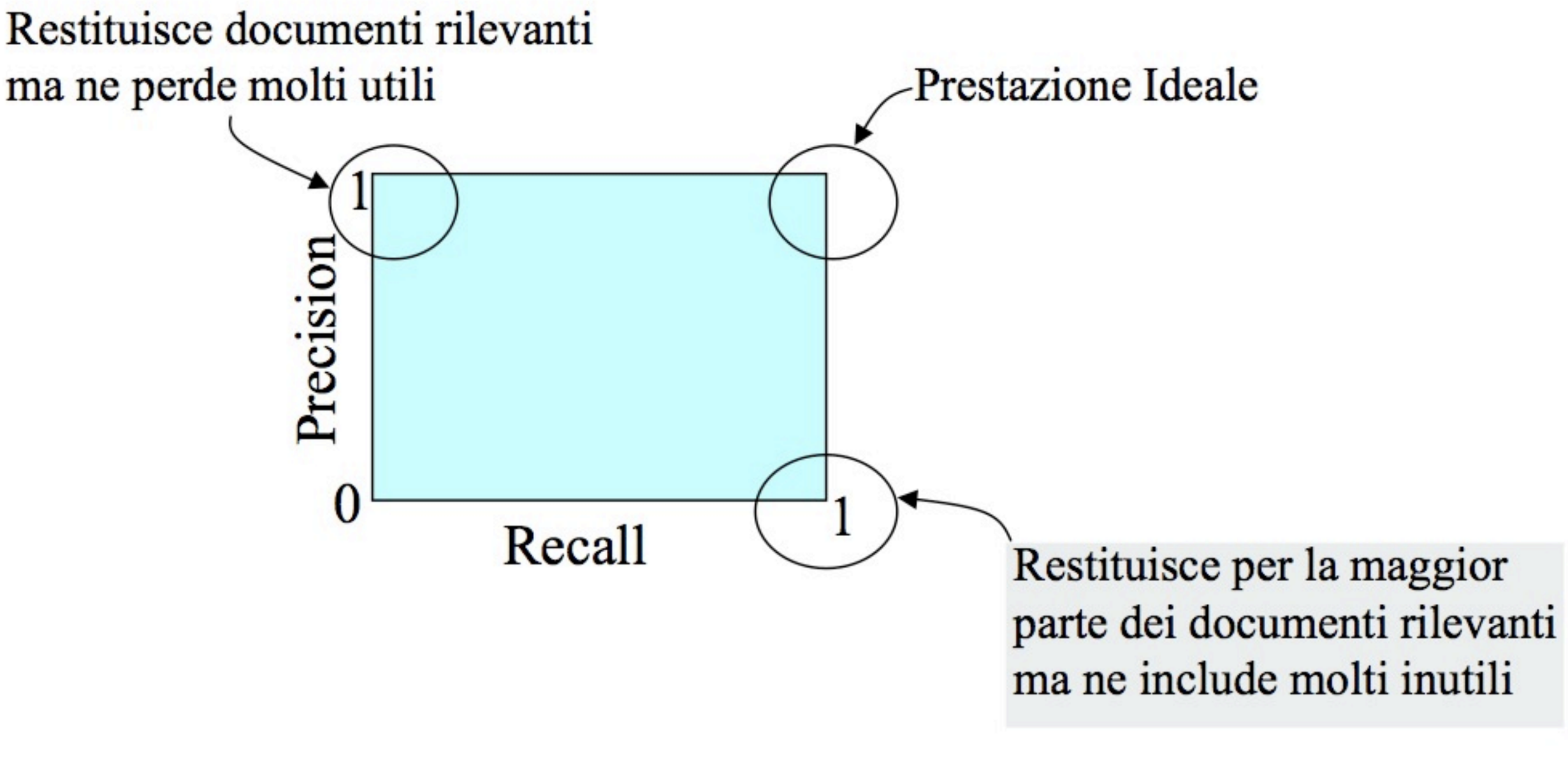
$$recall = \frac{Number\ of\ relevant\ documents\ retrieved}{Total\ number\ of\ relevant\ documents}$$

$$precision = \frac{Number\ of\ relevant\ documents\ retrieved}{Total\ number\ of\ documents\ retrieved}$$

Precision - Recall

- Precision
  - La capacità di trovare nei primi posti della lista ordinata di documenti, quelli altamente rilevanti
- Recall
  - La capacità di trovare tutti i documenti rilevanti nella collezione

Precision oppure Recall?



F-Measure

- Misura di prestazione che prende in considerazione sia recall che precision
- Media armonica tra recall e precision

$$F = \frac{2PR}{P+R} = \frac{2}{\frac{1}{R}\frac{1}{P}}$$

- Rispetto alla media artimetica è richiesto che entrambe siano alte affinchè la media armonica sia alta anch'essa.

F-Measure parametrizzata

- Una variante della F-Measure che permette di pesare l'importanza della precision sulla recall:

$$F = \frac{(1+\beta^2)PR}{\beta^2 P + R} = \frac{2}{\frac{\beta^2}{} \frac{1}{}}$$

- Il valore β controlla il trade-off:
  - β=1: peso identico di precision e recall
  - β>1: pesa la precision di più
  - β<1: pesa la recall di più

Fallout

- I problemi di precision e recall:
  - il numero dei documenti irrilevanti nella collezione non viene considerato
  - La recall non è definita se non c'è alcun documento rilevante nella collezione
  - La precision non è definita se non viene ritrovato alcun documento

$$Fallout = \frac{no.\,of\ nonrelevant\ items\ retrieved}{total\ no.\,of\ nonrelevant\ items\ in\ the\ collection}$$

Misure di rilevanza soggettive

- **Novelty Ratio:**
  - La percentuale di elementi ritrovati e giudicati rilevanti dall'utente e dei quali egli non era precedentemente al corrente.
    - cattura la capacità di trovare nuova informazione su un tema
- **Coverage Ratio:**
  - la percentuale di elementi rilevanti trovati rispetto al numero totale dei documenti di cui l'utente era al corrente.
    - Interessante nei casi in cui l'utente voglia localizzare i documenti che già aveva visto prima (e.g. un rapporto tecnico di qualche anno prima).

Altri fattori

- **User Effort**: lavoro richiesto all'utente per la formulazione della query, condurre la ricerca ed analizzare i risultati
- **Response Time**: intervallo temporale tra la ricezione della query e la presentazione dei risultati del sistema
- **Form of presentation**: influenza del formato di output dei risultati sulla capacità dell'utente di usare il materiale ritrovato
- **Collection coverage**: percentuale nella quale gli elementi rilevanti appartengono al corpus