Introduction to **Information Retrieval**

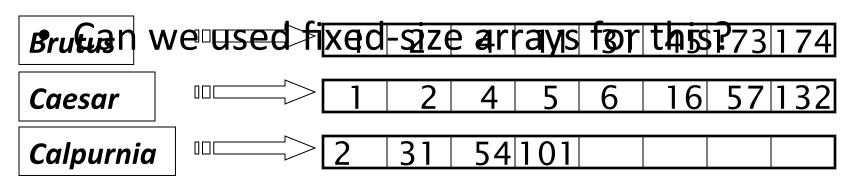
The Inverted Index

The key data structure underlying

modern IR

Inverted index

- For each term t, we must store a list of all documents that contain t.
 - Identify each doc by a docID, a document serial number



What happens if the word *Caesar* is added to document 14?

Inverted index

We need variable-size postings lists

Dictionary

- On disk, a continuous run of postings is normal and best
- In memory, can use linked lists or variable arrays

 Brutus
 Some tradeoffs in size/ease of insertion

 Caesar

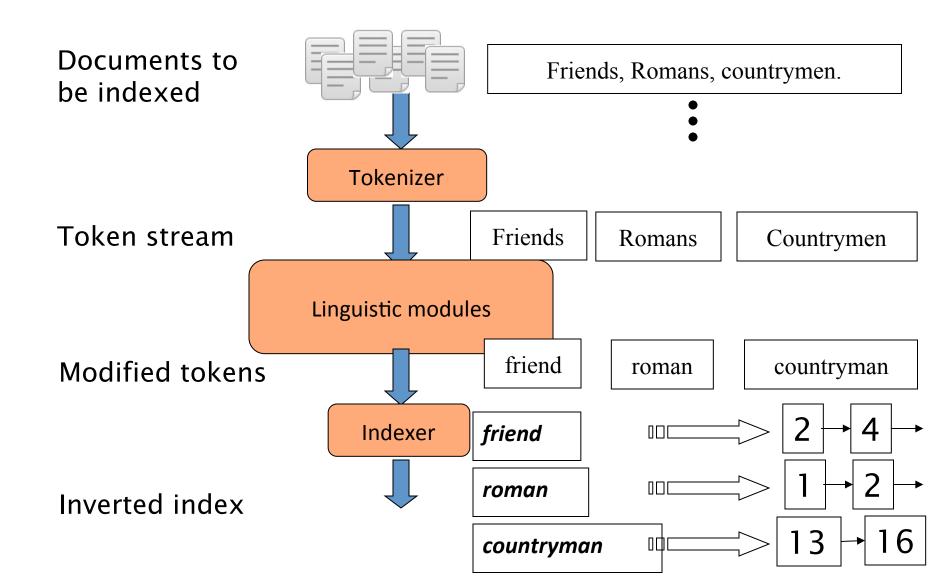
 Calpurnia

 2 31 54101

Sorted by docID (more later on why).

Postings

Inverted index construction



Initial stages of text processing

- Tokenization
 - Cut character sequence into word tokens
 - Deal with "John's", a state-of-the-art solution
- Normalization
 - Map text and query term to same form
 - You want *U.S.A.* and *USA* to match
- Stemming
 - We may wish different forms of a root to match
 - authorize, authorization
- Stop words
 - We may omit very common words (or not)
 - the, a, to, of

Indexer steps: Token sequence

Sequence of (Modified token, Document ID) pairs.

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

1		
did 1 enact 1 julius 1 caesar 1 I 1 1 was 1 killed 1 i' 1 the 1 capitol 1 brutus 1 killed 1 me 1 so 2 let 2 with 2 be 2 with 2 with 2 toaesar 2 the 2 noble 2 brutus 4 told 2 you 2 caesar 2 was 2	Term	docID
enact		
julius 1 caesar 1 I 1 1 was 1 killed 1 i' 1 the 1 capitol 1 brutus 1 killed 1 me 1 so 2 let 2 with 2 with 2 with 2 with 2 caesar 1 the 2 noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2	4.1.4.	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		
I	julius	
was 1 killed 1 i' 1 the 1 capitol 1 brutus 1 killed 1 me 1 so 2 let 2 with 2 caesar 2 the 2 noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2	caesar	
killed 1 i' 1 the 1 capitol 1 brutus 1 killed 1 me 1 so 2 let 2 it 2 with 2 with 2 caesar 2 the 2 noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2	I	
i' 1 the 1 capitol 1 brutus 1 killed 1 me 1 so 2 let 2 with 2 with 2 caesar 2 the 2 noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2	was	1
the 11 capitol 11 brutus 11 killed 11 me 11 so 22 let 22 it 22 with 22 with 22 noble 22 brutus 22 hath 22 told 22 you 22 caesar 22 was 22		1
capitol 1 brutus 1 killed 1 me 1 so 2 let 2 it 2 with 2 with 2 caesar 2 the 2 noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2	i'	
brutus 1 killed 1 me 1 so 2 let 2 it 2 with 2 with 2 caesar 2 the 2 noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2	the	
killed 11 me 11 so 22 let 22 it 22 with 22 with 22 the 22 brutus 22 brutus 22 brutus 42 told 22 you 22 caesar 22 was 22	capitol	
me 11 so 22 let 22 it 22 with 22 with 22 the 22 noble 22 brutus 22 hath 22 told 22 you 22 caesar 22 was 22	brutus	
me	killed	1
so 2 let 2 it 2 be 2 with 2 caesar 2 the 2 brutus 2 hath 2 told 2 you 2 caesar 2 ambitious 2	me	1
let 22 it 22 be 22 with 22 caesar 22 the 22 hath 22 hath 22 you 22 caesar 22 was 22 ambitious 22	so	2
it 22 be 22 with 22 caesar 22 the 22 noble 22 brutus 22 hath 22 you 22 caesar 22 was 22 ambitious 22	let	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	it	2
with 2 caesar 2 the 2 noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2 ambitious 2	be	2
caesar 22 the 22 noble 22 brutus 22 hath 22 told 22 you 22 caesar 22 was 22 ambitious 22	with	2
the 22 noble 22 brutus 22 hath 22 you 22 caesar 22 was 22 ambitious 22	caesar	2
noble 2 brutus 2 hath 2 told 2 you 2 caesar 2 was 2 ambitious 2	the	2
brutus 2 hath 2 told 2 you 2 caesar 2 was 2 ambitious 2	noble	2
hath 2 told 2 you 2 caesar 2 was 2 ambitious 2	brutus	2
told 2 you 2 caesar 2 was 2 ambitious 2	hath	2
you 2 caesar 2 was 2 ambitious 2	told	2
caesar 2 was 2 ambitious 2	you	2
was 2 ambitious 2	-	2
ambitious 2	was	2
	ambitious	2

Indexer steps: Sort

- Sort by terms
 - And then docID

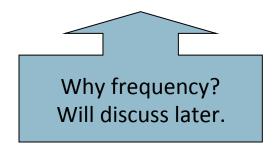


Term	docID
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
was	2
ambitious	2

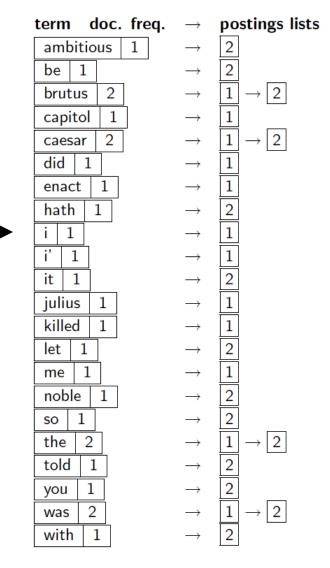
Term	docID
ambitious	2
be	2 1 2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	1 2 2 1 2 2 2 1 2 2 2 2 2
with	2

Indexer steps: Dictionary & Postings

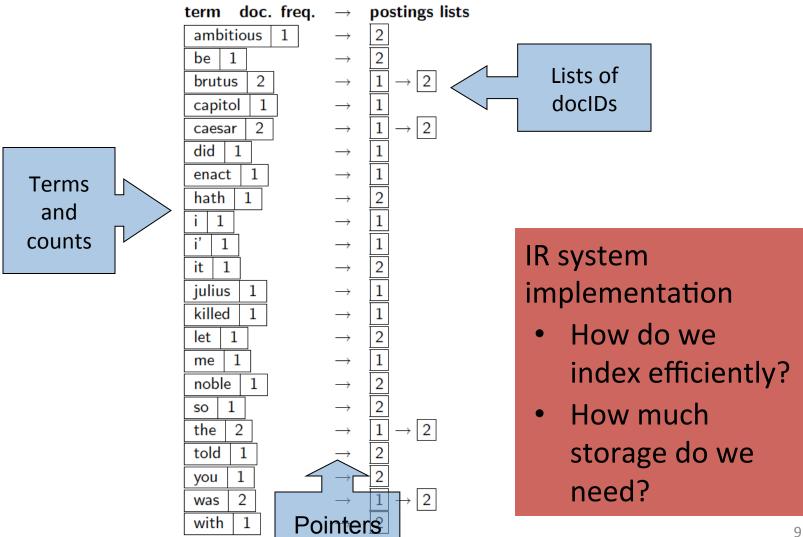
- Multiple term entries in a single document are merged.
- Split into Dictionary and Postings
- Doc. frequency information is added.







Where do we pay in storage?



Introduction to **Information Retrieval**

The Inverted Index

The key data structure underlying

modern IR