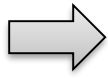
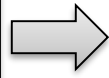


# Brief Information Retrieval Tutorial

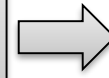
(with application in Software Engineering)



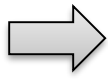
**Corpus  
Preprocessing  
Techniques**



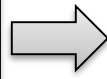
**Information  
Retrieval  
Techniques**



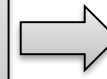
**Maintenance  
Tasks**



**Corpus  
Preprocessing  
Techniques**

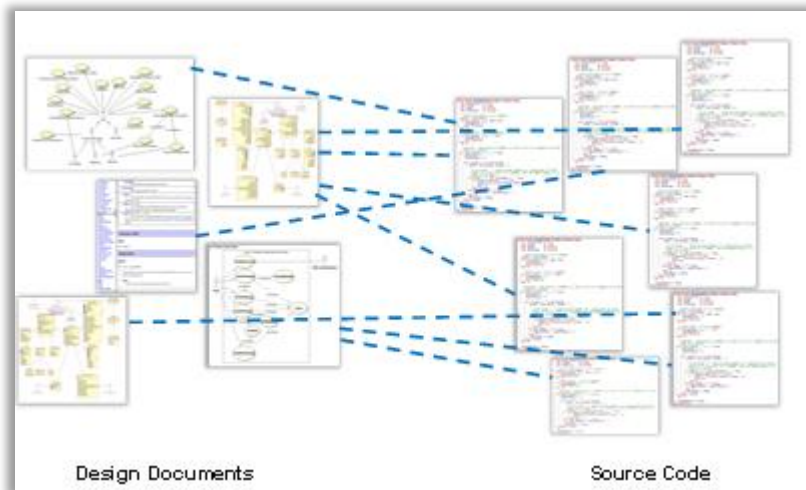


**Information  
Retrieval  
Techniques**

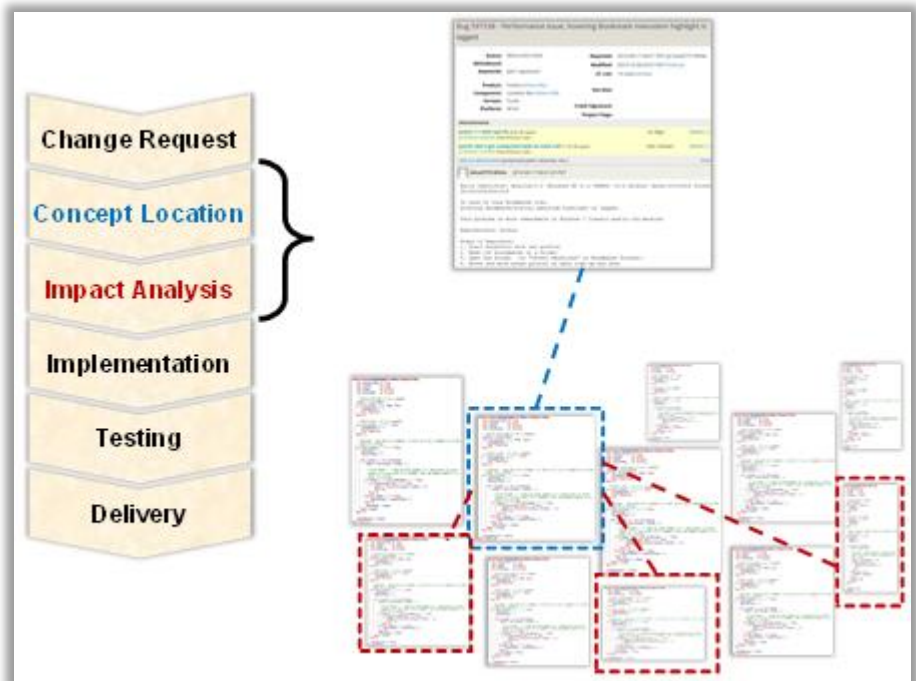


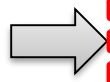
**Maintenance  
Tasks**

## Traceability Link Recovery

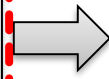


## Concept Location Impact Analysis

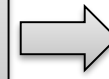




**Corpus  
Preprocessing  
Techniques**



**Information  
Retrieval  
Techniques**



**Maintenance  
Tasks**

```
Private Function CleanUpLine(ByVal sLine As String) As String
    Dim lQuoteCount As Long
    Dim lcount As Long
    Dim sChar As String
    Dim sPrevChar As String

    ' Starts with Rem it is a comment
    sLine = Trim(sLine)
    If Left(sLine, 3) = "Rem" Then
        CleanUpLine = ""
        Exit Function
    End If

    ' Starts with ' it is a comment
    If Left(sLine, 1) = "'" Then
        CleanUpLine = ""
        Exit Function
    End If

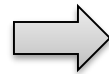
    ' Contains ' may end in a comment, so test if it is a comment or in the
    ' body of a string
    If InStr(sLine, "'") > 0 Then
        sPrevChar = ""
        lQuoteCount = 0

        For lcount = 1 To Len(sLine)
            sChar = Mid(sLine, lcount, 1)

            ' If we found " ' then an even number of " characters in front
            ' means it is the start of a comment, and odd number means it is
            ' part of a string
            If sChar = "" And sPrevChar = " " Then
                If lQuoteCount Mod 2 = 0 Then
                    sLine = Trim(Left(sLine, lcount - 1))
                    Exit For
                End If
            ElseIf sChar = "" Then
                lQuoteCount = lQuoteCount + 1
            End If
            sPrevChar = sChar
        Next lcount

        CleanUpLine = sLine
    End Function
```

(AST)  
Parser



```
synchronized void print(TestResult result,  
                           long runTime) throws IOException
```

```
{
```

```
    //standard header format  
    printHeader(runTime);
```

```
    printErrors(result);  
    printFailures(result);  
    printFooter(result);
```

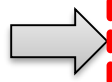
```
}
```

```
synchronized void printHeader(long runTime)  
                           throws IOException
```

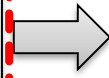
```
{
```

```
    ...
```

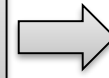
```
}
```



**Corpus  
Preprocessing  
Techniques**



**Information  
Retrieval  
Techniques**



**Maintenance  
Tasks**

**Remove  
Special  
Characters**

```
synchronized void print(TestMethod  
result, long runTime) throws  
IOException
```

```
{
```

```
    //standard header format  
    printHeader(runTime);
```

```
    printErrors(result);  
    printFailures(result);  
    printFooter(result);
```

```
}
```

```
synchronized void printHeader(long  
runTime) throws IOException
```

```
{
```

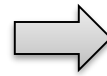
```
    ...
```

```
}
```

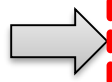
```
synchronized void print TestResult  
result long runTime  throws  
IOException
```

```
standard header format  
printHeader runTime
```

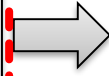
```
printErrors result  
printFailures result  
printFooter result
```



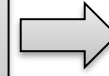
```
synchronized void printHeader long  
runTime  
throws IOException
```



**Corpus  
Preprocessing  
Techniques**

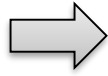


**Information  
Retrieval  
Techniques**



**Maintenance  
Tasks**

**Remove  
Special  
Characters**



**Split  
Identifiers**

```
synchronized void print TestResult  
result long runTime throws  
IOException
```

```
standard header format  
printHeader runTime
```

```
printErrors result  
printFailures result  
printFooter result
```

```
synchronized void printHeader long  
runTime  
throws IOException
```

```
synchronized void print test result  
result long run time throws  
io exception
```

```
standard header format  
print header run time
```

```
print errors result  
print failures result  
print footer result
```

```
synchronized void print header long  
run time  
throws io exception
```

# Identifier Splitting Algorithms

---

## Original Identifier

---

**userId**

**setGID**

**print\_file2device**

**SSLCertificate**

**MINstring**

**USERID**

**currentsize**

**tolocale**

**imitating**

**DEFMASKBit**

---

# Identifier Splitting Algorithms

Original Identifier	Camel Case
userId	user Id
setGID	set GID
print_file2device	print file 2 device
SSLCertificate	SSL Certificate
MINstring	MI Nstring
USERID	USERID
currentsize	currentsize
tolocale	tolocale
imitating	imitating
DEFMASKBit	DEFMASK Bit



# Identifier Splitting Algorithms

Original Identifier	Camel Case
userId	user Id
setGID	set GID
print_file2device	print file 2 device
SSLCertificate	SSL Certificate
MINstring	MI Nstring
USERID	USERID
currentsize	currentsize
tolocale	tolocale
imitating	imitating
DEFMASKBit	DEFMASK Bit

Handles underscore  
and digits

Fails at mixed cases

Fails at same case identifiers

# Identifier Splitting Algorithms

Original Identifier	Camel Case	Samurai*
userId	user Id	user Id
setGID	set GID	set GID
print_file2device	print file 2 device	print file 2 device
SSLCertificate	SSL Certificate	SSL Certificate
MINstring	MI Nstring	MIN string
USERID	USERID	USER ID
currentsize	currentsize	current size
tolocale	tolocale	tol ocal e
imitating	imitating	imi ta ting
DEFMASKBit	DEFMASK Bit	DEF MASK Bit

\*[Enslen'09]

# Identifier Splitting Algorithms

Original Identifier	Camel Case	Samurai*
userId	user Id	user Id
setGID	set GID	set GID
print_file2device	print file 2 device	print file 2 c
SSLCertificate	SSL Certificate	SSL Certificate
MINstring	MI Nstring	MIN string
USERID	USERID	USER ID
currentsize	currentsize	current size
tolocale	tolocale	tol ocal e
imitating	imitating	imi ta ting
DEFMASKBit	DEFMASK Bit	DEF MASK Bit

Splits some cases  
that CamelCase  
cannot

Oversplits

# Identifier Splitting Algorithms

Original Identifier	Camel Case	Samurai*
userId	user Id	user Id
setGID	set GID	set GID

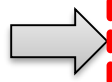
Splits some cases  
that CamelCase

Choose **meaningful identifier names**

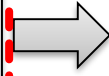
- Improves code **readability**

Use **consistent naming conventions**

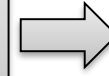
- promotes **shared code ownership** (XP)
- Improves **search** results



**Corpus  
Preprocessing  
Techniques**

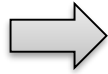


**Information  
Retrieval  
Techniques**



**Maintenance  
Tasks**

**Remove  
Special  
Characters**



**Split  
Identifiers**

```
synchronized void print TestResult  
result long runTime throws  
IOException
```

```
standard header format  
printHeader runTime
```

```
printErrors result  
printFailures result  
printFooter result
```

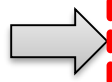
```
synchronized void printHeader long  
runTime  
throws IOException
```

```
synchronized void print test result  
result long run time throws  
io exception
```

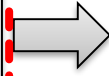
```
standard header format  
print header run time
```

```
print errors result  
print failures result  
print footer result
```

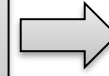
```
synchronized void print header long  
run time  
throws io exception
```



**Corpus  
Preprocessing  
Techniques**

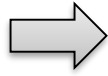


**Information  
Retrieval  
Techniques**

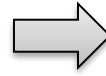


**Maintenance  
Tasks**

**Remove  
Special  
Characters**



**Split  
Identifiers**



**Remove  
Stop  
Words**

`synchronized void print test result  
result long run time throws  
io exception`

`standard header format  
print header run time`

`print errors result  
print failures result  
print footer result`

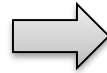
`synchronized void print header long  
run time  
throws io exception`

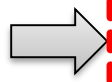
`print test result  
result run time  
io exception`

`standard header format  
print header run time`

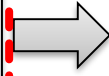
`print errors result  
print failures result  
print footer result`

`print header  
run time  
io exception`





**Corpus  
Preprocessing  
Techniques**

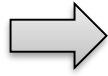


**Information  
Retrieval  
Techniques**

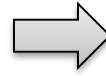


**Maintenance  
Tasks**

**Remove  
Special  
Characters**



**Split  
Identifiers**



**Remove  
Stop  
Words**



**Stem**

```
print test result
result run time
io exception
```

```
standard header format
print header run time
```

```
print errors result
print failures result
print footer result
```

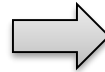
```
print header
run time
io exception
```

```
print test result
result run time
io exception
```

```
standard head format
print head run time
```

```
print error result
print fail result
print foot result
```

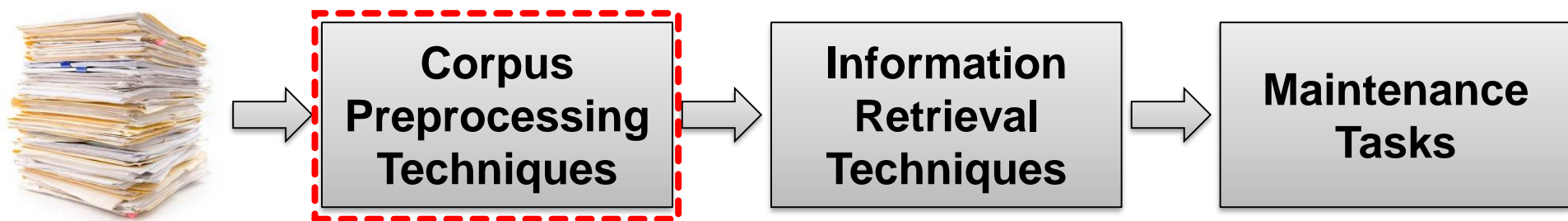
```
print head
run time
io exception
```



# Stemming Errors

- If stemmer is too aggressive
  - organization  $\Rightarrow$  organ
  - police  $\Rightarrow$  policy
  - army  $\Rightarrow$  arm
  - executive  $\Rightarrow$  execute





## Summary of Preprocessing (Extract only the “meaningful” words from the documents in the corpus)

```
synchronized void print(TestResult result,  
                        long runTime) throws IOException
```

```
{
```

```
    //standard header format
```

```
    printHeader(runTime);
```

```
    printErrors(result);
```

```
    printFailures(result);
```

```
    printFooter(result);
```

```
}
```

```
synchronized void printHeader(long runTime)  
                        throws IOException
```

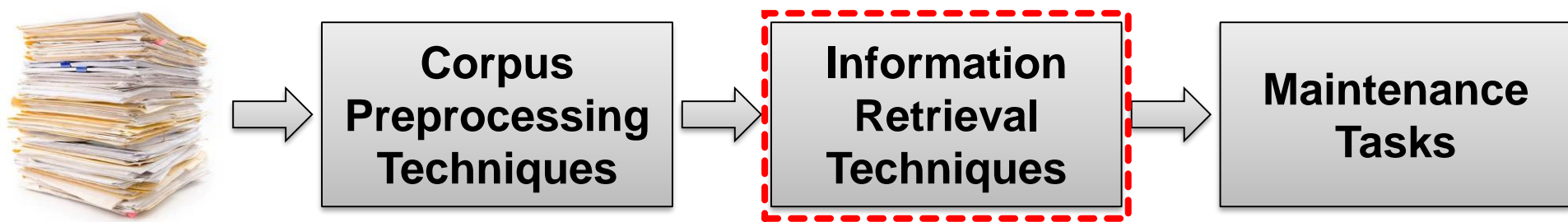
```
{
```

```
print test result  
result run time  
io exception
```

```
standard head format  
print head run time
```

```
print error result  
print fail result  
print foot result
```

```
print head  
run time  
io exception
```

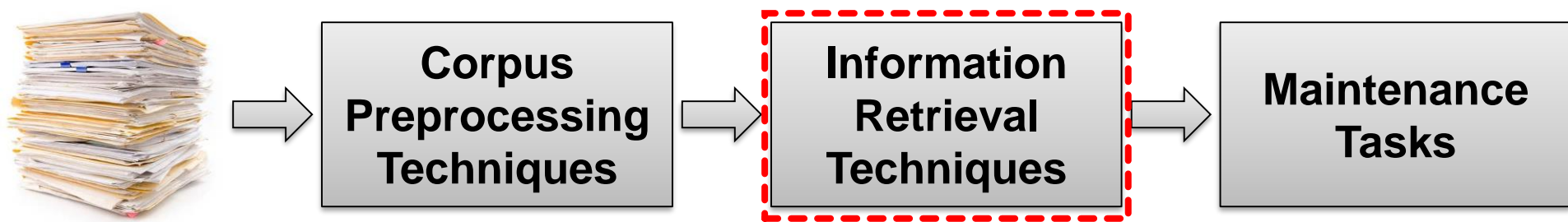


$m_1$ : print test result  
result run time  
io exception

standard head format  
print head run time

print error result  
print fail result  
print foot result

$m_2$ : print head  
run time  
io exception



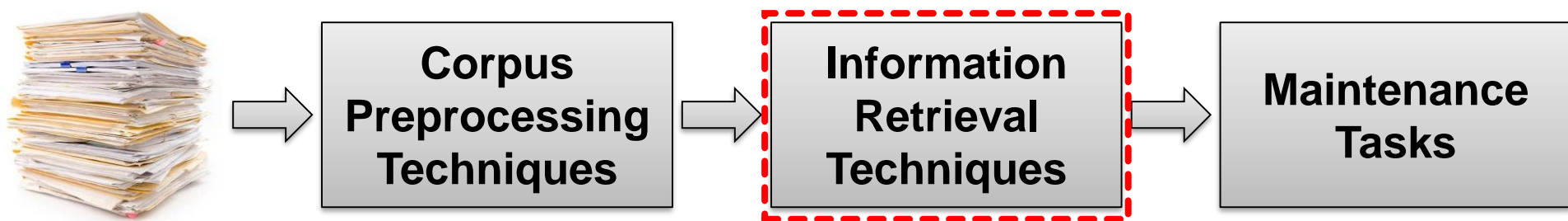
### Term-by-Document Matrix

$m_1$ : print test result  
result run time  
io exception

standard head format  
print head run time

print error result  
print fail result  
print foot result

$m_2$ : print head  
run time  
io exception



### Term-by-Document Matrix

$m_1$ : print test result  
result run time  
io exception

standard head format  
print head run time

print error result  
print fail result  
print foot result

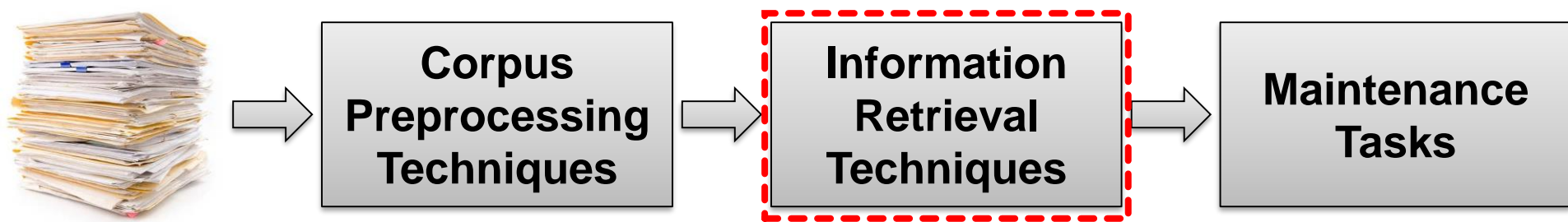
$m_2$ : print head  
run time  
io exception

**Unique Terms**

	print	test	result	time	...
$m_1$	5	1	5	2	
$m_2$	1	0	0	1	
...	...	...	...	...	
$m_n$	...	...	...	...	

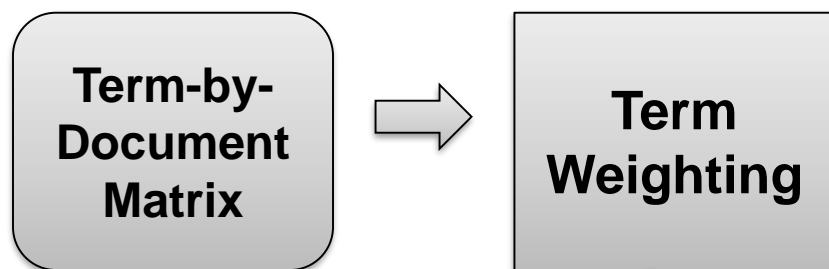
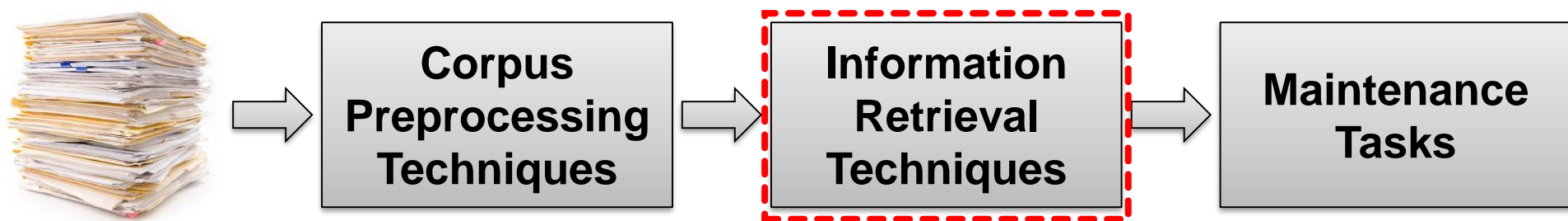
**Methods**

Each document is represented  
as a **vector of terms**

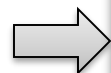


**Term-by-Document Matrix**

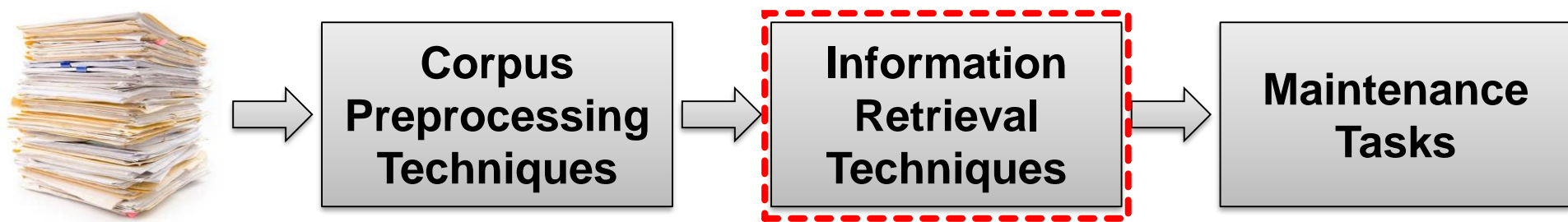
	print	test	result	...
$m_1$	5	1	3	...
$m_2$	1	0	0	...



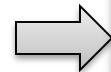
	print	test	result	...
$m_1$	5	1	3	...
$m_2$	1	0	0	...



boolean  
tf  
tf-idf  
 $\log(\text{tf}+1)$   
tf-entropy



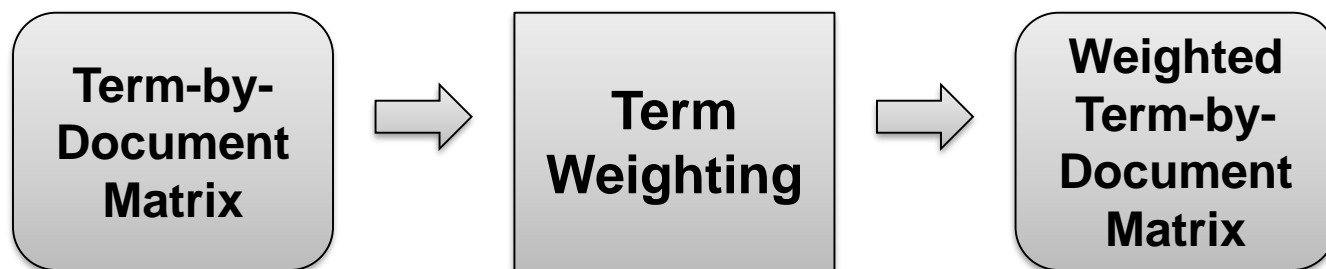
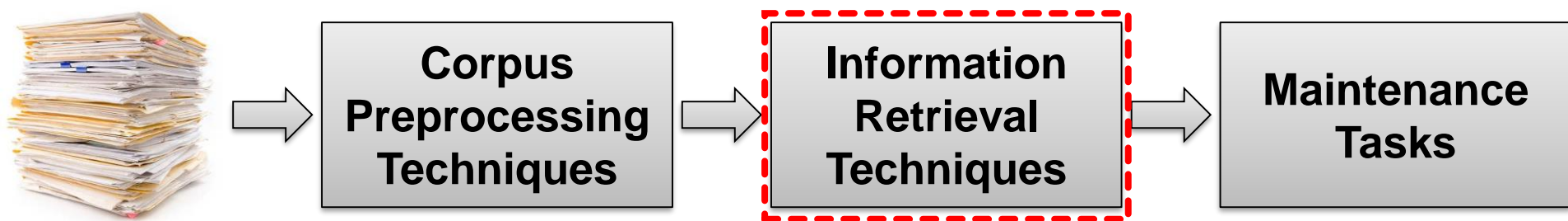
	print	test	result	...
$m_1$	5	1	3	...
$m_2$	1	0	0	...



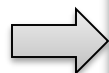
boolean  
tf  
tf-idf  
 $\log(\text{tf}+1)$   
tf-entropy



	print	test	result	...
$m_1$	0.78	0.2	0.9	...
$m_2$	0.45	0	0	...



	print	test	result	...
$m_1$	5	1	3	...



boolean  
tf  
tf-idf  
 $\log(\text{tf}+1)$

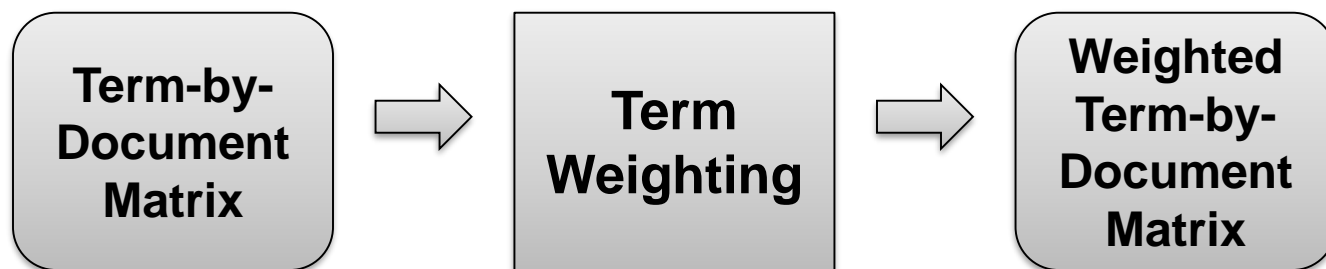
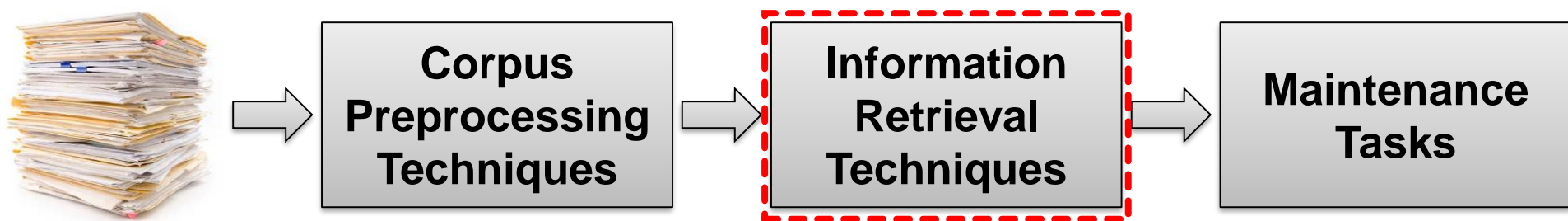


	print	test	result	...
$m_1$	0.78	0.2	0.9	...

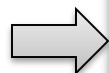
Each term in the document vector is **“weighted”** based on its **“importance”** to the **document** (method) and **corpus** (collection of documents).

**Why?**





	print	test	result	...
$m_1$	5	1	3	...
	1	0	0	...



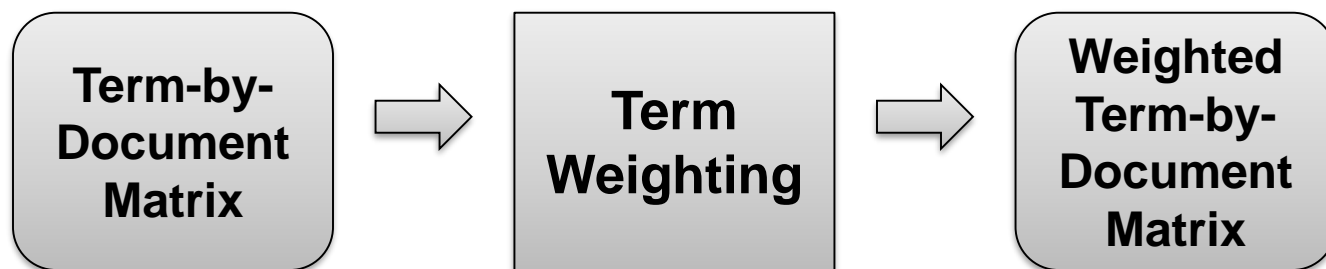
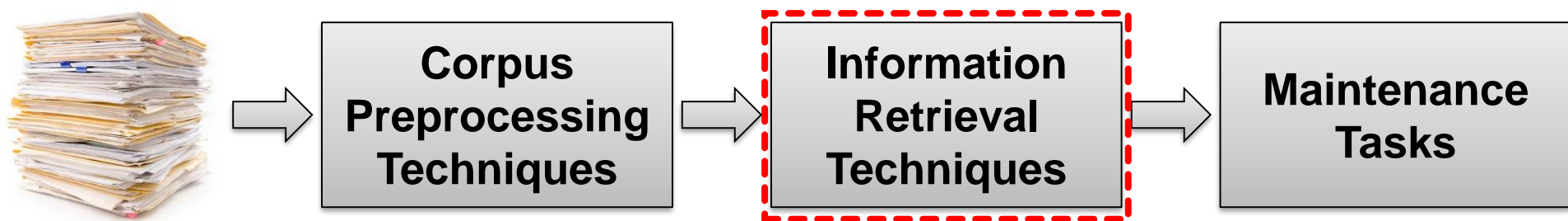
boolean  
tf  
tf-idf  
 $\log(\text{tf}+1)$



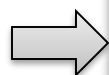
	print	test	result	...
$m_1$	0.78	0.2	0.9	...
	0.45	0	0	...

Each term in the document vector is **“weighted”** based on its **“importance”** to the **document** (method) and **corpus** (collection of documents).

In different contexts, some terms are more important than others



	print	test	result	...
$m_1$	5	1	3	...



boolean  
tf  
tf-idf  
 $\log(\text{tf}+1)$



	print	test	result	...
$m_1$	0.78	0.2	0.9	...

Each term in the document vector is “weighted” based on its “importance” to the document (method) and corpus (collection of documents).

In different contexts, some terms are more important than others

# Term Weights

Term Frequency (TF)

+

Inverse Document Frequency (IDF)

=

TF-IDF

# Assumption for Using Term Weights...

- Terms that appear in many *different* documents are ...

# Assumption for Using Term Weights...

- Terms that appear in many *different* documents are
  - less indicative of the overall topic
  - carry less “meaning”
- Term weights represent an indication of a term’s discriminative power

# Inverse Document Frequency (IDF) Example

$Doc_1$

$Doc_2$

$Doc_3$

...

...

...

...

...

...

...

...

...

...

...

...

...

...

...

$Doc_{10K}$

- In a corpus of 10,000 documents

$Doc_1$	a a a
$Doc_2$	a
$Doc_3$	a a
...	a a
...	a a a
...	a
...	a
...	a a
...	a a a
...	a a a
...	a
...	a a
...	a a
...	a a a
...	a
...	a a a
.....	.....
...	a
$Doc_{10k}$	a

10,000 documents

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be

$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$



$Doc_1$	a a a
$Doc_2$	a
$Doc_3$	a a
...	a a
...	a a a
...	a
...	a
...	a a
...	a a a
...	a a a
...	a
...	a a
...	a a
...	a a a
...	a
...	a a a
...	...
...	a
$Doc_{10K}$	a

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
 
$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$
- Term b (appears in 5,000 documents)

$Doc_1$	a a a	b b
$Doc_2$	a	
$Doc_3$	a a	b
...	a a	
...	a a a	b b b b
...	a	
...	a	b b
...	a a	
...	a a a	b
...	a a a	
...	a	b b b
...	a a	
...	a a	b b
...	a a a	
...	a	b b b b
...	a a a	
.....		
...	a	b b
$Doc_{10K}$	a	

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
 
$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$
- Term b (appears in 5,000 documents)

$Doc_1$	a a a	b b
$Doc_2$	a	
$Doc_3$	a a	b
...	a a	
...	a a a	b b b b
...	a	
...	a	b b
...	a a	
...	a a a	b
...	a a a	
...	a	b b b
...	a a	
...	a a	b b
...	a a a	
...	a	b b b b
...	a a a	
.....		
...	a	b b
$Doc_{10K}$	a	

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
- Term b (appears in 5,000 documents)

$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$

$$idf_b = \log \left( \frac{10,000}{5,000} \right) = 0.69$$

$Doc_1$	a a a	b b
$Doc_2$	a	
$Doc_3$	a a	b
...	a a	
...	a a a	b b b b
...	a	
...	a	b b
...	a a	
...	a a a	b
...	a a a	
...	a	b b b
...	a a	
...	a a	b b
...	a a a	
...	a	b b b b
...	a a a	
.....		
...	a	b b
$Doc_{10K}$	a	

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
 
$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$
- Term b (appears in 5,000 documents)
 
$$idf_b = \log \left( \frac{10,000}{5,000} \right) = 0.69$$
- Term c (appears in 20 documents)

$Doc_1$	a a a	b b	c c
$Doc_2$	a		
$Doc_3$	a a	b	c
...	a a		
...	a a a	b b b b	
...	a		
...	a	b b	c c c
...	a a		
...	a a a	b	
...	a a a		
...	a	b b b	
...	a a		c c c c
...	a a	b b	
...	a a a		
...	a	b b b b	
...	a a a		
.....			
...	a	b b	
$Doc_{10K}$	a		c

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
 
$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$
- Term b (appears in 5,000 documents)
 
$$idf_b = \log \left( \frac{10,000}{5,000} \right) = 0.69$$
- Term c (appears in 20 documents)

$Doc_1$	a a a	b b	c c
$Doc_2$	a		
$Doc_3$	a a	b	c
...	a a		
...	a a a	b b b b	
...	a		
...	a	b b	c c c
...	a a		
...	a a a	b	
...	a a a		
...	a	b b b	
...	a a		c c c c
...	a a	b b	
...	a a a		
...	a	b b b b	
...	a a a		
.....			
...	a	b b	
$Doc_{10K}$	a		c

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)

- Its *discriminative power* (IDF) will be

$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$

- Term b (appears in 5,000 documents)

$$idf_b = \log \left( \frac{10,000}{5,000} \right) = 0.69$$

- Term c (appears in 20 documents)

$$idf_c = \log \left( \frac{10,000}{20} \right) = 6.21$$

$Doc_1$	a a a	b b	c c
$Doc_2$	a		
$Doc_3$	a a	b	c
...	a a		
...	a a a	b b b b	
...	a		
...	a	b b	c c c
...	a a		
...	a a a	b	
...	a a a		
...	a	b b b	
...	a a		c c c c
...	a a	b b	
...	a a a		
...	a	b b b b	
...	a a a		
.....			
...	a	b b	
$Doc_{10K}$	a		c

- In a **corpus of 10,000 documents**
- Term a:
  - **Appears in 10,000 documents** (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
 
$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$
- Term b (**appears in 5,000 documents**)
 
$$idf_b = \log \left( \frac{10,000}{5,000} \right) = 0.69$$
- Term c (**appears in 20 documents**)
 
$$idf_c = \log \left( \frac{10,000}{20} \right) = 6.21$$
- Term d (**appears in 1 document**)

$Doc_1$	a a a	b b	c c	
$Doc_2$	a			d d d
$Doc_3$	a a	b	c	
...	a a			
...	a a a	b b b b		
...	a			
...	a	b b	c c c	
...	a a			
...	a a a	b		
...	a a a			
...	a	b b b		
...	a a		c c c c	
...	a a	b b		
...	a a a			
...	a	b b b b		
...	a a a			
.....				
...	a	b b		
$Doc_{10K}$	a		c	

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
 
$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$
- Term b (appears in 5,000 documents)
 
$$idf_b = \log \left( \frac{10,000}{5,000} \right) = 0.69$$
- Term c (appears in 20 documents)
 
$$idf_c = \log \left( \frac{10,000}{20} \right) = 6.21$$
- Term d (appears in 1 document)



$Doc_1$	a a a	b b	c c	
$Doc_2$	a			d d d
$Doc_3$	a a	b	c	
...	a a			
...	a a a	b b b b		
...	a			
...	a	b b	c c c	
...	a a			
...	a a a	b		
...	a a a			
...	a	b b b		
...	a a		c c c c	
...	a a	b b		
...	a a a			
...	a	b b b b		
...	a a a			
.....				
...	a	b b		
$Doc_{10K}$	a		c	

- In a corpus of 10,000 documents
- Term a:
  - Appears in 10,000 documents (it can appear multiple times in the same document)
  - Its *discriminative* power (IDF) will be
 
$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$
- Term b (appears in 5,000 documents)
 
$$idf_b = \log \left( \frac{10,000}{5,000} \right) = 0.69$$
- Term c (appears in 20 documents)
 
$$idf_c = \log \left( \frac{10,000}{20} \right) = 6.21$$
- Term d (appears in 1 document)
 
$$idf_d = \log \left( \frac{10,000}{1} \right) = 9.21$$

$Doc_1$	a a a	b b	c c
$Doc_2$	a		d d d
$Doc_3$	a a	b	c
...	a a		
...	a a a	b b b b	
...	a		
...	a	b b	c c c
...	a a		
...	a a a	b	

■ In a corpus of 10,000 documents

■ Term a:

■ Appears in 10,000 documents (it can appear multiple times in the same document)

■ Its *discriminative* power (IDF) will be

$$idf_a = \log \left( \frac{10,000}{10,000} \right) = 0$$

Inverse Document Frequency (IDF) provides:

- low values for common words and
- high values for rare words (with discriminative power)

...	a	b b	
$Doc_{10K}$	a		c

■ Term d (appears in 1 document)

$$idf_d = \log \left( \frac{10,000}{1} \right) = 9.21$$

# Computing TF-IDF: An Example

- In a corpus of 10,000 documents



$\text{Doc}_1$	a a a      b b            c
$\text{Doc}_2$	_____
$\text{Doc}_3$	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	_____
...	.....
...	_____
$\text{Doc}_{10K}$	

- In a corpus of 10,000 documents
- Term a

aaa

- 3 times in  $Doc_1$

$Doc_1$	a a a	b b	c
$Doc_2$			
$Doc_3$			
...			
...	a a a		
...			
...	a		
...			
...			
...			
...			
...	a a		
...			
...			
...			
...			
...			
...			
...			
...			
...			
...			
$Doc_{10k}$			

50 documents

- In a corpus of 10,000 documents
- Term a
  - Appears 3 times in  $Doc_1$
- Appears in 50 documents



50 documents

- $$tf_{Doc_1,a} = \frac{3}{3} = 1$$

<i>Doc</i> <sub>1</sub>	a a a	b b	c
<i>Doc</i> <sub>2</sub>			
<i>Doc</i> <sub>3</sub>			
...			
...	a a a		
...			
...	a		
...			
...			
...			
...			
...	a a		
...			
...			
...			
...			
...			
...			
...			
...			
...			
...			
<i>Doc</i> <sub>10k</sub>			

■ In a corpus of 10,000 documents

■ Term a

■ Appears 3 times in *Doc*<sub>1</sub>

■ Its normalized term frequency (TF) in *Doc*<sub>1</sub> is:

$$tf_{Doc_1, a} = \frac{3}{3} = 1$$

■ Appears in 50 documents

■ Its *discriminative* power (IDF) will be

$$idf_a = \log \left( \frac{10,000}{50} \right) = 5.29$$

50 documents

- $$tf_{Doc_1,a} = \frac{3}{3} = 1$$

- $$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$

- $\text{tfidf}_{\text{Doc}_1, a} = \text{tf}_{\text{Doc}_1, a} * \text{idf}_a = 1 * 5.29 = 5.29$

“weight” (“importance”) of term  $a$  in  $Doc_1$  is 5.29 (i.e., the tf-idf)



[illegible]

- In a corpus of 10,000 documents
- Term a
  - Appears 3 times in  $Doc_1$
  - Its normalized term frequency (TF) in  $Doc_1$  is:
$$tf_{Doc_1,a} = \frac{3}{3} = 1$$
  - Appears in 50 documents
  - Its *discriminative* power (IDF) will be
$$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$
  - $tfidf_{Doc_1,a} = tf_{Doc_1,a} * idf_a = 1 * 5.29 = 5.29$
- Term b

$Doc_1$	a a a	b b	c
$Doc_2$			
$Doc_3$			
...			
...	a a a		
...			
...	a		
...			
...			
...			
...			
...	a a		
...			
...			
...			
...			
...			
...			
...			
$Doc_{10K}$			

■ In a corpus of 10,000 documents

■ Term a

■ Appears 3 times in  $Doc_1$

■ Its normalized term frequency (TF) in  $Doc_1$  is:

$$tf_{Doc_1,a} = \frac{3}{3} = 1$$

■ Appears in 50 documents

■ Its *discriminative* power (IDF) will be

$$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$

■  $tfidf_{Doc_1,a} = tf_{Doc_1,a} * idf_a = 1 * 5.29 = 5.29$

■ Term b

■ Appears 2 times in  $Doc_1 \Rightarrow tf_{Doc_1,b} = \frac{2}{3} = 0.66$

2 times

$Doc_1$	a a a	b b	c
$Doc_2$		b b b	
$Doc_3$		b	
...			
...	a a a	b b b b	
...		b	
...	a	b b	
...		b b b	
...		b	
...			
...		b b b	
...	a a		
...		b b	
...		b b b	
...		b b b b	
...		b	
.....			
...		b b	
$Doc_{10K}$			

1,300 documents

- In a corpus of 10,000 documents

- Term a

- Appears 3 times in  $Doc_1$

- Its normalized term frequency (TF) in  $Doc_1$  is:

$$tf_{Doc_1,a} = \frac{3}{3} = 1$$

- Appears in 50 documents

- Its *discriminative* power (IDF) will be

$$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$

- $tfidf_{Doc_1,a} = tf_{Doc_1,a} * idf_a = 1 * 5.29 = 5.29$

- Term b

- Appears 2 times in  $Doc_1 \Rightarrow tf_{Doc_1,b} = \frac{2}{3} = 0.66$

- Appears in 1,300 documents  $\Rightarrow idf_b = \log\left(\frac{10,000}{1,300}\right) = 2.04$

$Doc_1$	a a a	b b	c
$Doc_2$		b b b	
$Doc_3$		b	
...			
...	a a a	b b b b	
...		b	
...	a	b b	
...		b b b	
...		b	
...			
...		b b b	
...	a a		
...		b b	
...		b b b	
...		b b b b	
...		b	
.....			
...		b b	
$Doc_{10K}$			

1,300 documents

■ In a corpus of 10,000 documents

■ Term a

■ Appears 3 times in  $Doc_1$

■ Its normalized term frequency (TF) in  $Doc_1$  is:

$$tf_{Doc_1,a} = \frac{3}{3} = 1$$

■ Appears in 50 documents

■ Its *discriminative* power (IDF) will be

$$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$

■  $tfidf_{Doc_1,a} = tf_{Doc_1,a} * idf_a = 1 * 5.29 = 5.29$

■ Term b

■ Appears 2 times in  $Doc_1 \Rightarrow tf_{Doc_1,b} = \frac{2}{3} = 0.66$

■ Appears in 1,300 documents  $\Rightarrow idf_b = \log\left(\frac{10,000}{1,300}\right) = 2.04$

■  $tfidf_{Doc_1,b} = tf_{Doc_1,b} * idf_b = 0.66 * 2.04 = 1.36$



$Doc_1$	a a a	b b	c
$Doc_2$		b b b	
$Doc_3$		b	
...			
...	a a a	b b b b	
...		b	
...	a	b b	
...		b b b	
...		b	
...			
...		b b b	
...	a a		
...		b b	
...		b b b	
...		b b b b	
...		b	
.....			
...		b b	
$Doc_{10K}$			

■ In a corpus of 10,000 documents

■ Term a

■ Appears 3 times in  $Doc_1$

■ Its normalized term frequency (TF) in  $Doc_1$  is:

$$tf_{Doc_1,a} = \frac{3}{3} = 1$$

■ Appears in 50 documents

■ Its *discriminative* power (IDF) will be

$$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$

■  $tfidf_{Doc_1,a} = tf_{Doc_1,a} * idf_a = 1 * 5.29 = 5.29$

■ Term b

■ Appears 2 times in  $Doc_1 \Rightarrow tf_{Doc_1,b} = \frac{2}{3} = 0.66$

■ Appears in 1,300 documents  $\Rightarrow idf_b = \log\left(\frac{10,000}{1,300}\right) = 2.04$

■  $tfidf_{Doc_1,b} = tf_{Doc_1,b} * idf_b = 0.66 * 2.04 = 1.36$

■ Term c

$Doc_1$	a a a	b b <span style="border: 1px dashed purple; padding: 2px;">c</span>
$Doc_2$		b b b
$Doc_3$		b
...		
...	a a a	b b b b
...		b
...	a	b b
...		b b b
...		b
...		
...		b b b
...	a a	
...		b b
...		b b b
...		b b b b
...		b
.....		
...		b b
$Doc_{10K}$		

■ In a corpus of 10,000 documents

■ Term a

■ Appears 3 times in  $Doc_1$

■ Its normalized term frequency (TF) in  $Doc_1$  is:

$$tf_{Doc_1,a} = \frac{3}{3} = 1$$

■ Appears in 50 documents

■ Its *discriminative* power (IDF) will be

$$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$

■  $tfidf_{Doc_1,a} = tf_{Doc_1,a} * idf_a = 1 * 5.29 = 5.29$

■ Term b

■ Appears 2 times in  $Doc_1 \Rightarrow tf_{Doc_1,b} = \frac{2}{3} = 0.66$

■ Appears in 1,300 documents  $\Rightarrow idf_b = \log\left(\frac{10,000}{1,300}\right) = 2.04$

■  $tfidf_{Doc_1,b} = tf_{Doc_1,b} * idf_b = 0.66 * 2.04 = 1.36$

■ Term c

■ Appears 1 time in  $Doc_1 \Rightarrow tf_{Doc_1,c} = \frac{1}{3} = 0.33$

$Doc_1$	a a a	b b	c
$Doc_2$		b b b	
$Doc_3$		b	c c c
...			c c
...	a a a	b b b b	
...		b	
...	a	b b	c c c
...		b b b	
...		b	
...			c c
...		b b b	
...	a a		c c c c
...		b b	
...		b b b	
...		b b b b	
...		b	
.....			
...		b b	
$Doc_{10K}$			c

250 documents

■ In a corpus of 10,000 documents

■ Term a

■ Appears 3 times in  $Doc_1$

■ Its normalized term frequency (TF) in  $Doc_1$  is:

$$tf_{Doc_1,a} = \frac{3}{3} = 1$$

■ Appears in 50 documents

■ Its *discriminative* power (IDF) will be

$$idf_a = \log\left(\frac{10,000}{50}\right) = 5.29$$

■  $tfidf_{Doc_1,a} = tf_{Doc_1,a} * idf_a = 1 * 5.29 = 5.29$

■ Term b

■ Appears 2 times in  $Doc_1 \Rightarrow tf_{Doc_1,b} = \frac{2}{3} = 0.66$

■ Appears in 1,300 documents  $\Rightarrow idf_b = \log\left(\frac{10,000}{1,300}\right) = 2.04$

■  $tfidf_{Doc_1,b} = tf_{Doc_1,b} * idf_b = 0.66 * 2.04 = 1.36$

■ Term c

■ Appears 1 time in  $Doc_1 \Rightarrow tf_{Doc_1,c} = \frac{1}{3} = 0.33$

■ Appears in 250 documents  $\Rightarrow idf_c = \log\left(\frac{10,000}{250}\right) = 3.68$

$Doc_1$	a a a	b b	c
$Doc_2$		b b b	
$Doc_3$		b	c c c
...			c c
...	a a a	b b b b	
...		b	
...	a	b b	c c c
...		b b b	
...		b	
...			c c
...		b b b	
...	a a		c c c c
...		b b	
...		b b b	
...		b b b b	
...		b	
.....			
...		b b	
$Doc_{10K}$			c

250 documents

■ In a corpus of 10,000 documents

■ Term a

■ Appears 3 times in  $Doc_1$

■ Its normalized term frequency (TF) in  $Doc_1$  is:

$$tf_{Doc_1, a} = \frac{3}{3} = 1$$

■ Appears in 50 documents

■ Its *discriminative* power (IDF) will be

$$idf_a = \log \left( \frac{10,000}{50} \right) = 5.29$$

■  $tfidf_{Doc_1, a} = tf_{Doc_1, a} * idf_a = 1 * 5.29 = 5.29$

■ Term b

■ Appears 2 times in  $Doc_1 \Rightarrow tf_{Doc_1, b} = \frac{2}{3} = 0.66$

■ Appears in 1,300 documents  $\Rightarrow idf_b = \log \left( \frac{10,000}{1,300} \right) = 2.04$

■  $tfidf_{Doc_1, b} = tf_{Doc_1, b} * idf_b = 0.66 * 2.04 = 1.36$

■ Term c

■ Appears 1 time in  $Doc_1 \Rightarrow tf_{Doc_1, c} = \frac{1}{3} = 0.33$

■ Appears in 250 documents  $\Rightarrow idf_c = \log \left( \frac{10,000}{250} \right) = 3.68$

■  $tfidf_{Doc_1, c} = tf_{Doc_1, c} * idf_c = 0.33 * 3.68 = 1.22$



# IDF Formulas

(FYI – Not Required for exams)

$df_i$  = “document frequency” of term  $i$   
= number of documents containing term  $i$

$N$  = total number of documents in the corpus

$idf_i = \log_2 \left( \frac{N}{df_i} \right)$   
= inverse document frequency of term  $i$

- Log used to dampen the effect relative to  $TF$

# TF-IDF Formulas

(FYI – Not Required for exams)

$$w_{ik} = tf_{ik} * idf_k = tf_{ik} * \log \left( \frac{N}{n_k} \right)$$

$w_{ik}$  = “weight” (“importance”) of term  $k$  in doc.  $D_i$

$T_k$  = term  $k$  in document  $D_i$

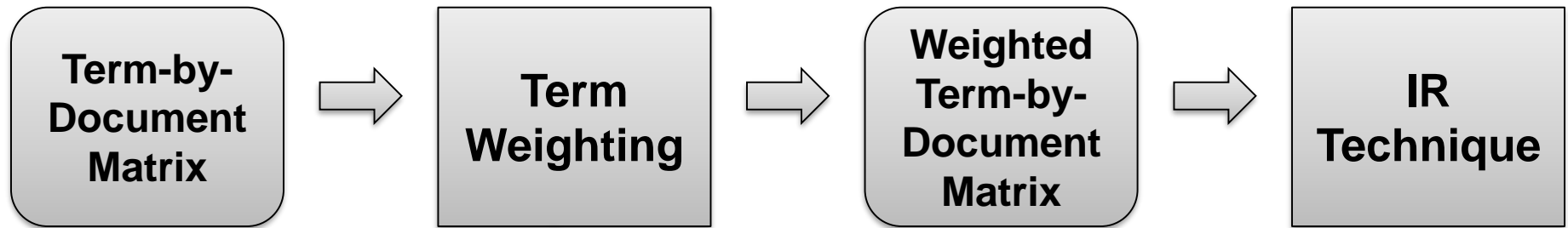
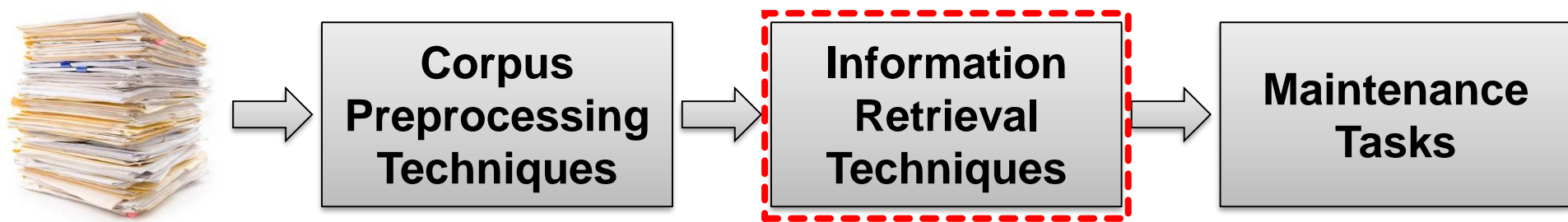
$tf_{ik}$  = (normalized) frequency of term  $T_k$  in doc.  $D_i$

$idf_k$  = inverse document frequency of term  $T_k$

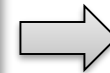
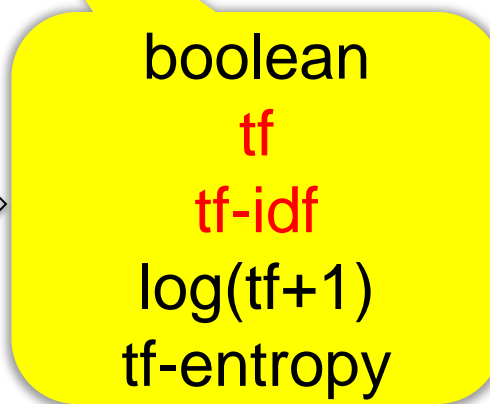
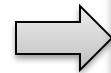
$$= \log \left( \frac{N}{n_k} \right)$$

$N$  = total number of documents in the corpus

$n_k$  = the number of documents in the corpus that contain  $T_k$

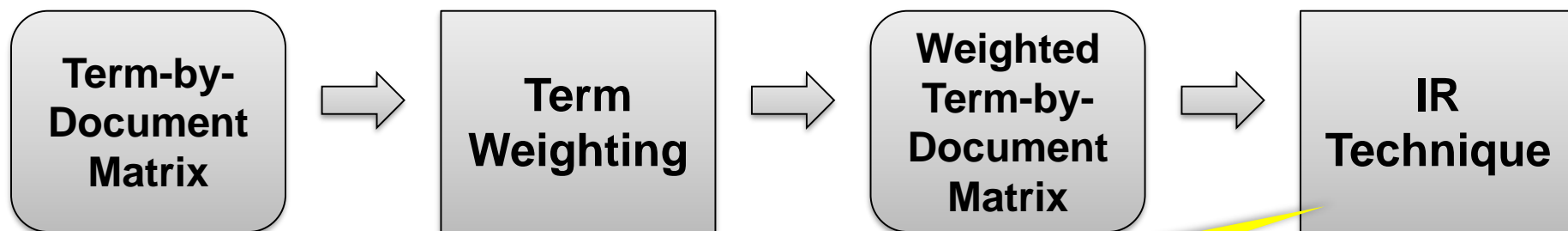
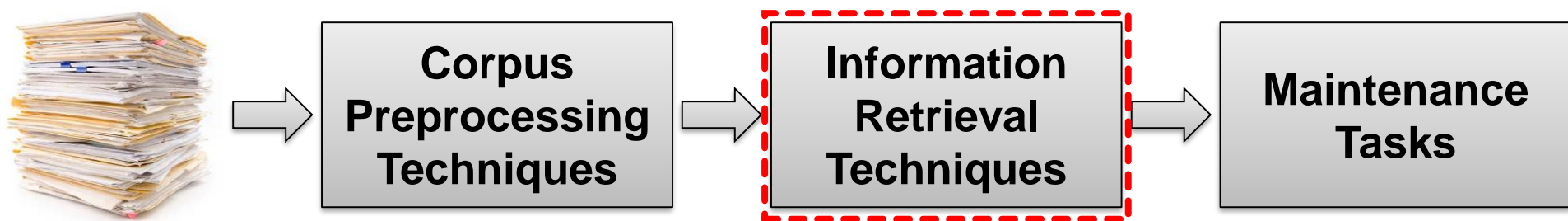


	print	test	result	...
$m_1$	5	1	3	...
$m_2$	1	0	0	...

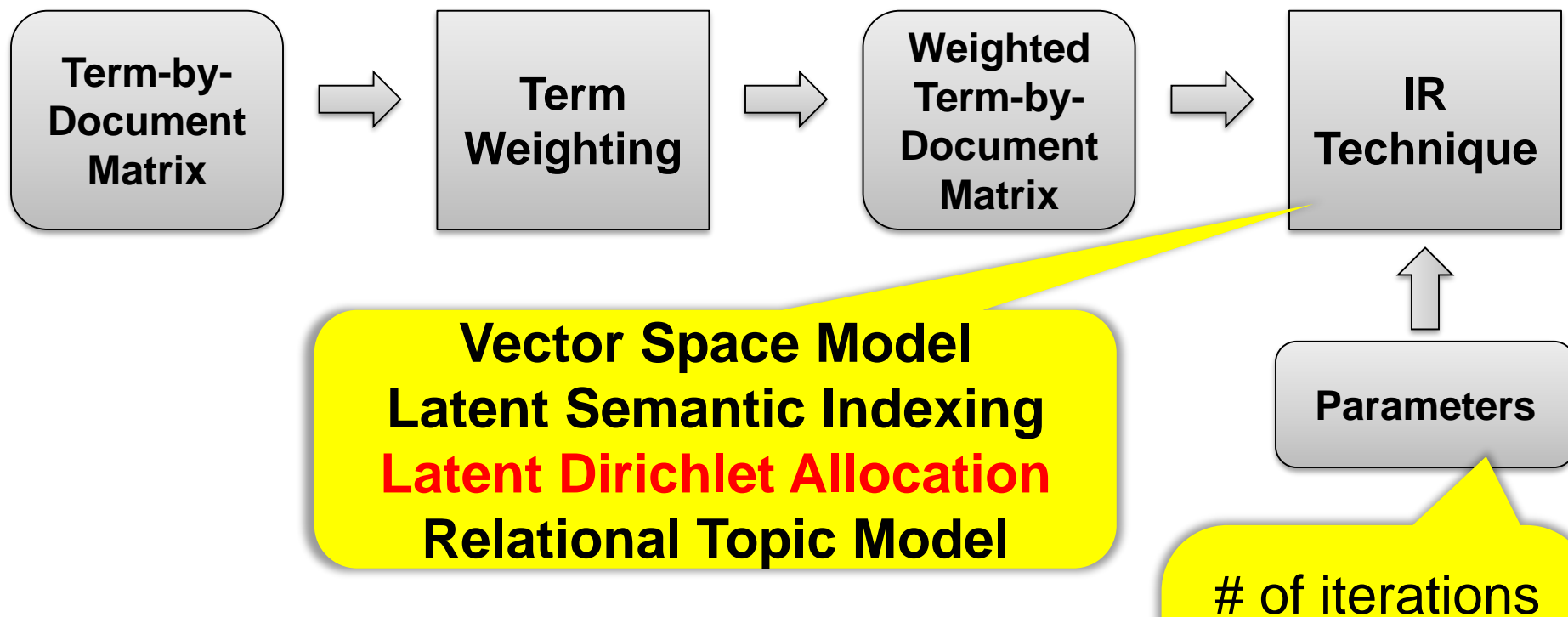
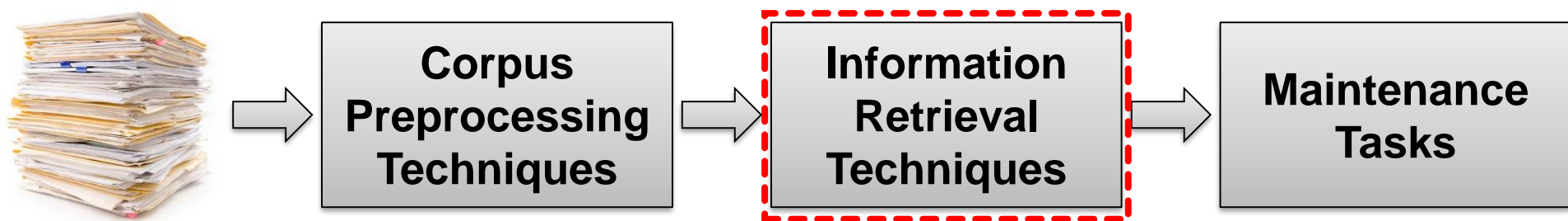


	print	test	result	...
$m_1$	0.78	0.2	0.9	...
$m_2$	0.45	0	0	...





**Vector Space Model**  
**Latent Semantic Indexing**  
**Latent Dirichlet Allocation**  
**Relational Topic Model**



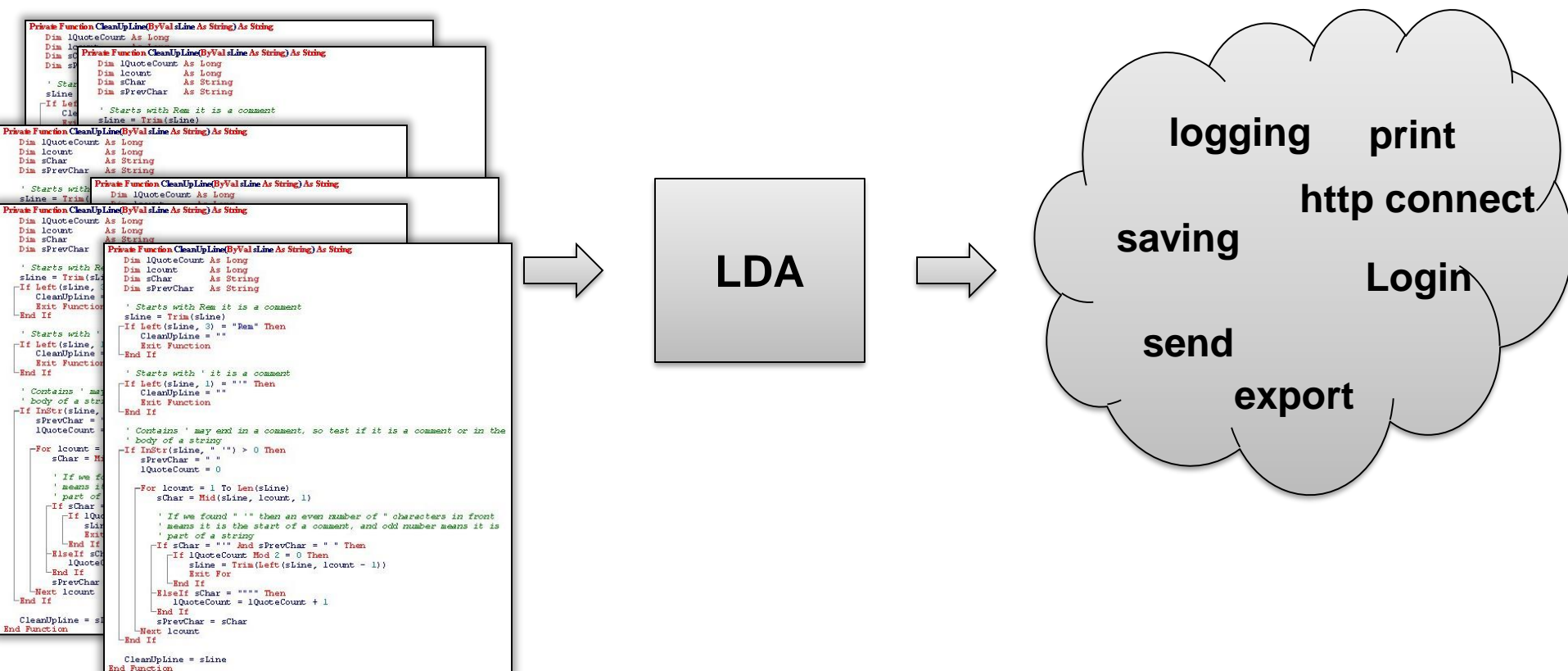
The choice of IR techniques and their configuration matters (i.e., affects the results)

# Latent Dirichlet Allocation (LDA)\*

- Topic model that generates the distribution of latent topics from textual documents

\*[Blei'03]

- Topic model that generates the distribution of latent topics from textual documents



# Example of Topics

Topic 1	
Image	0.09
Player	0.09
Path	0.07
Data	0.04
Audio	0.03
...	...

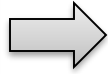
Topic 2	
android	0.13
activity	0.09
intent	0.08
thread	0.05
runtime	0.03
...	...

Topic 3	
database	0.11
string	0.11
table	0.07
sqlite	0.05
id	0.03
...	...

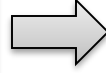
## **Software Artifacts**

6 documents

**Software  
Artifacts**



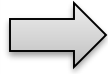
**Preprocessing**



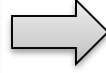
**Term-by-Document  
Matrix**

6 documents

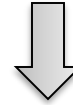
**Software  
Artifacts**



**Preprocessing**



**Term-by-Document  
Matrix**

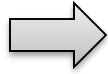


**LDA**

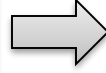
6 documents



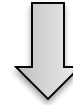
**Software  
Artifacts**



**Preprocessing**

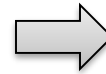


**Term-by-Document  
Matrix**



6 documents

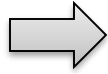
**Input  
Parameters**



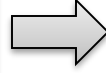
**LDA**

We want "3 topics"

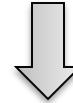
**Software  
Artifacts**



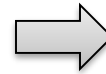
**Preprocessing**



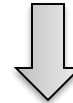
**Term-by-Document  
Matrix**



**Input  
Parameters**



**LDA**

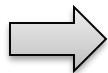


We want "3 topics"

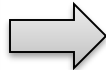
**Topic by Documents**

6 documents

**Software  
Artifacts**

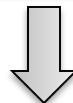


**Preprocessing**



**Term-by-Document  
Matrix**

6 documents

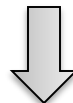


**Input  
Parameters**



**LDA**

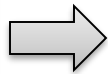
We want "3 topics"



**Topic by Documents**

	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1

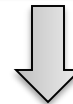
**Software  
Artifacts**



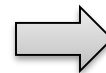
**Preprocessing**



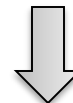
**Term-by-Document  
Matrix**



**Input  
Parameters**



**LDA**

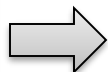


**Topic by Documents**

Probability that document  
is related to topic

	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1

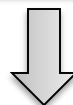
**Software  
Artifacts**



**Preprocessing**



**Term-by-Document  
Matrix**



**Input  
Parameters**



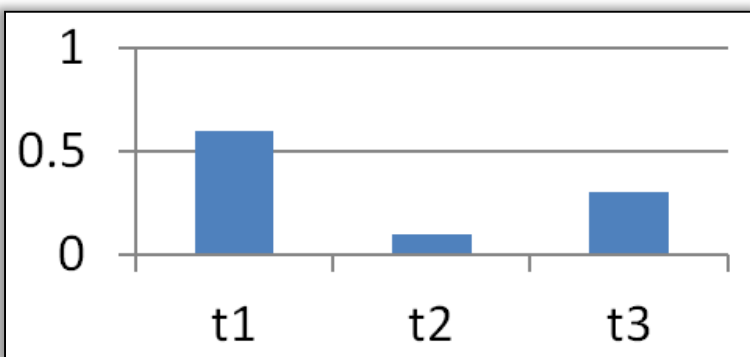
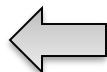
**LDA**



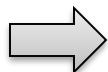
**Topic by Documents**

**t1 t2 t3**

<b>d1</b>	0.6	0.1	0.3
<b>d2</b>	0.25	0.65	0.1
<b>d3</b>	0.7	0.2	0.1
<b>d4</b>	0.05	0.05	0.9
<b>d5</b>	0.3	0.5	0.2
<b>d6</b>	0.15	0.75	0.1



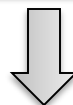
**Software  
Artifacts**



**Preprocessing**



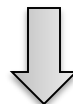
**Term-by-Document  
Matrix**



**Input  
Parameters**



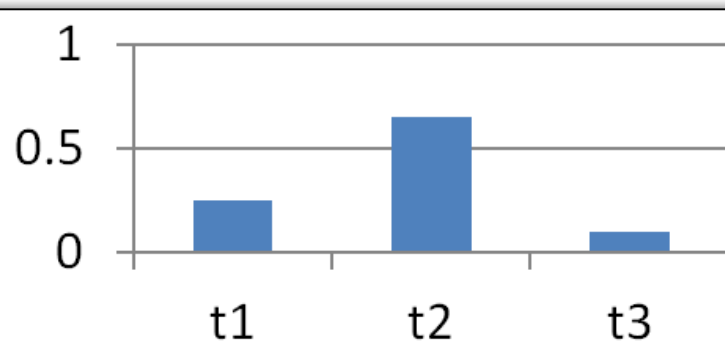
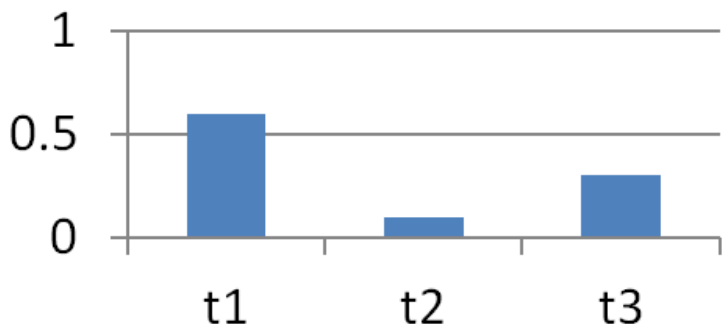
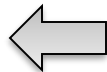
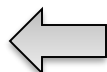
**LDA**



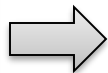
**Topic by Documents**

**t1 t2 t3**

<b>d1</b>	0.6	0.1	0.3
<b>d2</b>	0.25	0.65	0.1
<b>d3</b>	0.7	0.2	0.1
<b>d4</b>	0.05	0.05	0.9
<b>d5</b>	0.3	0.5	0.2
<b>d6</b>	0.15	0.75	0.1



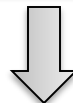
**Software  
Artifacts**



**Preprocessing**



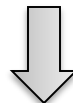
**Term-by-Document  
Matrix**



**Input  
Parameters**



**LDA**



**Topic by Documents**

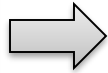
Compute document to  
document similarity

Compute query to  
document similarity

“Cluster” documents by  
topics

	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1

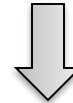
**Software  
Artifacts**



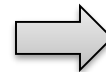
**Preprocessing**



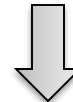
**Term-by-Document  
Matrix**



**Input  
Parameters**



**LDA**

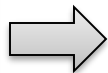


**Topic by Documents**

	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1



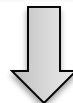
**Software  
Artifacts**



**Preprocessing**



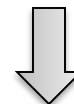
**Term-by-Document  
Matrix**



**Input  
Parameters**



**LDA**



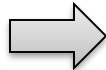
**Topic by Documents**

	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1

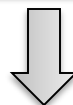
**Software  
Artifacts**



**Preprocessing**



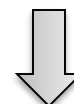
**Term-by-Document  
Matrix**



**Input  
Parameters**



**LDA**



**Words to Topic**

**Topic by Documents**

	w1	w2	w3	w4	...	wn
t1	0.2	0.1	0.4	0.15	...	0.1
t2	0.04	0.07	0.01	0.1	...	0.04
t3	0.03	0	0.17	0.12	...	0.02

	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1

Software  
Artifacts

Preprocessing

Term-by-Document  
Matrix

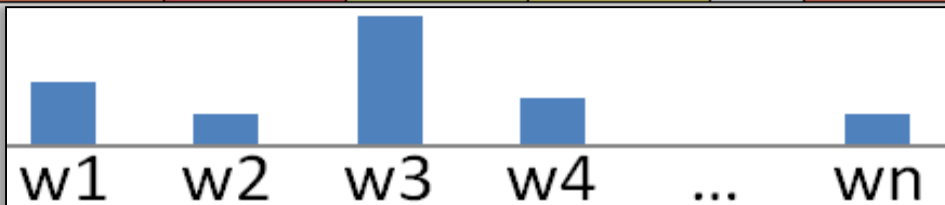
Input  
Parameters

LDA

Words to Topic

Topic by Documents

	w1	w2	w3	w4	...	wn
t1	0.2	0.1	0.4	0.15	...	0.1
t2	0.04	0.07	0.01	0.1	...	0.04
t3	0.03	0	0.17	0.12	...	0.02



	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1

Software  
Artifacts

Preprocessing

Term-by-Document  
Matrix

Input  
Parameters

LDA

Words to Topic

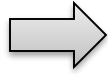
Topic by Documents

	w1	w2	w3	w4	...	wn
t1	0.2	0.1	0.4	0.15	...	0.1
t2	0.04	0.07	0.01	0.1	...	0.04
t3	0.03	0	0.17	0.12	...	0.02

Generate topic "labels"  
(top 5 words)

	t1	t2	t3
d1	0.6	0.1	0.3
d2	0.25	0.65	0.1
d3	0.7	0.2	0.1
d4	0.05	0.05	0.9
d5	0.3	0.5	0.2
d6	0.15	0.75	0.1

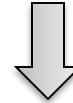
**Software  
Artifacts**



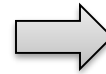
**Preprocessing**



**Term-by-Document  
Matrix**



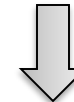
**Input  
Parameters**



**LDA**

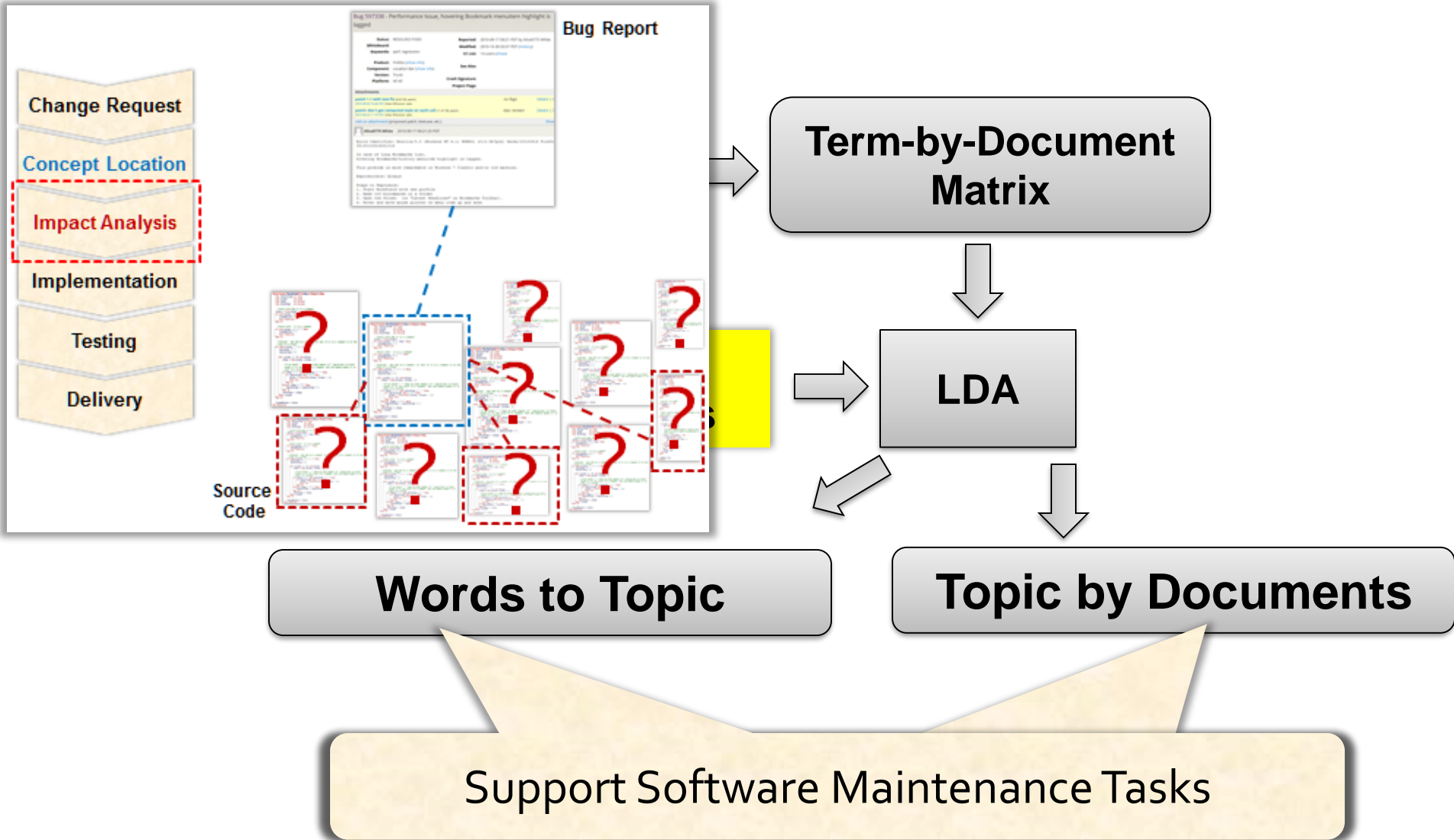


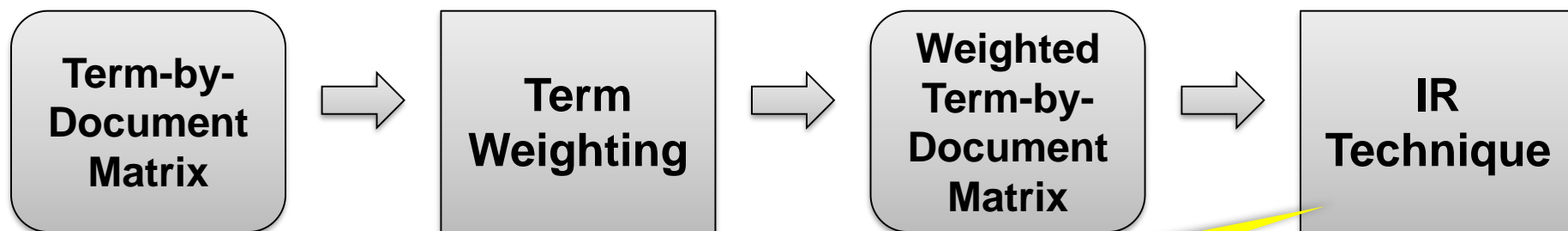
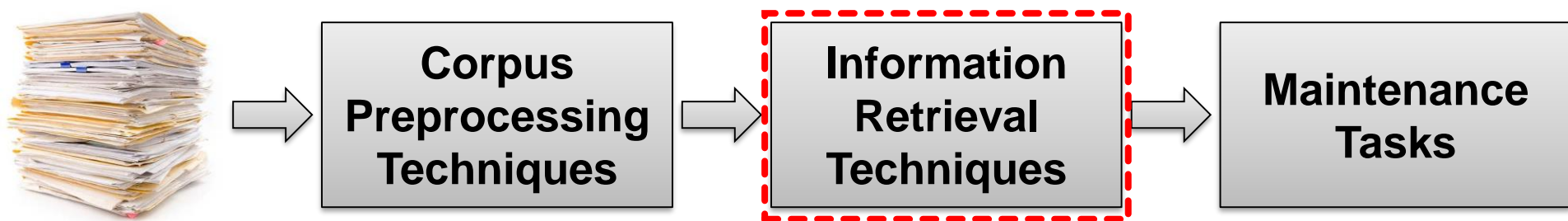
**Words to Topic**



**Topic by Documents**

Support Software Maintenance Tasks





**Vector Space Model**  
**Latent Semantic Indexing**  
**Latent Dirichlet Allocation**  
**Relational Topic Model**

# Graphic Representation Vector Space Model

$D_1: T_1 T_1$        $T_2 T_2 T_2$        $T_3 T_3 T_3 T_3 T_3$   
 $D_2: T_1 T_1 T_1$        $T_2 T_2 T_2 T_2 T_2 T_2 T_2$        $T_3$   
Search query  $Q:$        $T_3 T_3$



# Graphic Representation Vector Space Model

Example:

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

$$D_2 = 3T_1 + 7T_2 + T_3$$

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

Search query  $Q$  :

$D_1: T_1 T_1$	$T_2 T_2 T_2$	$T_3 T_3 T_3 T_3 T_3$
$D_2: T_1 T_1 T_1$	$T_2 T_2 T_2 T_2 T_2 T_2 T_2$	$T_3$
		$T_3 T_3$

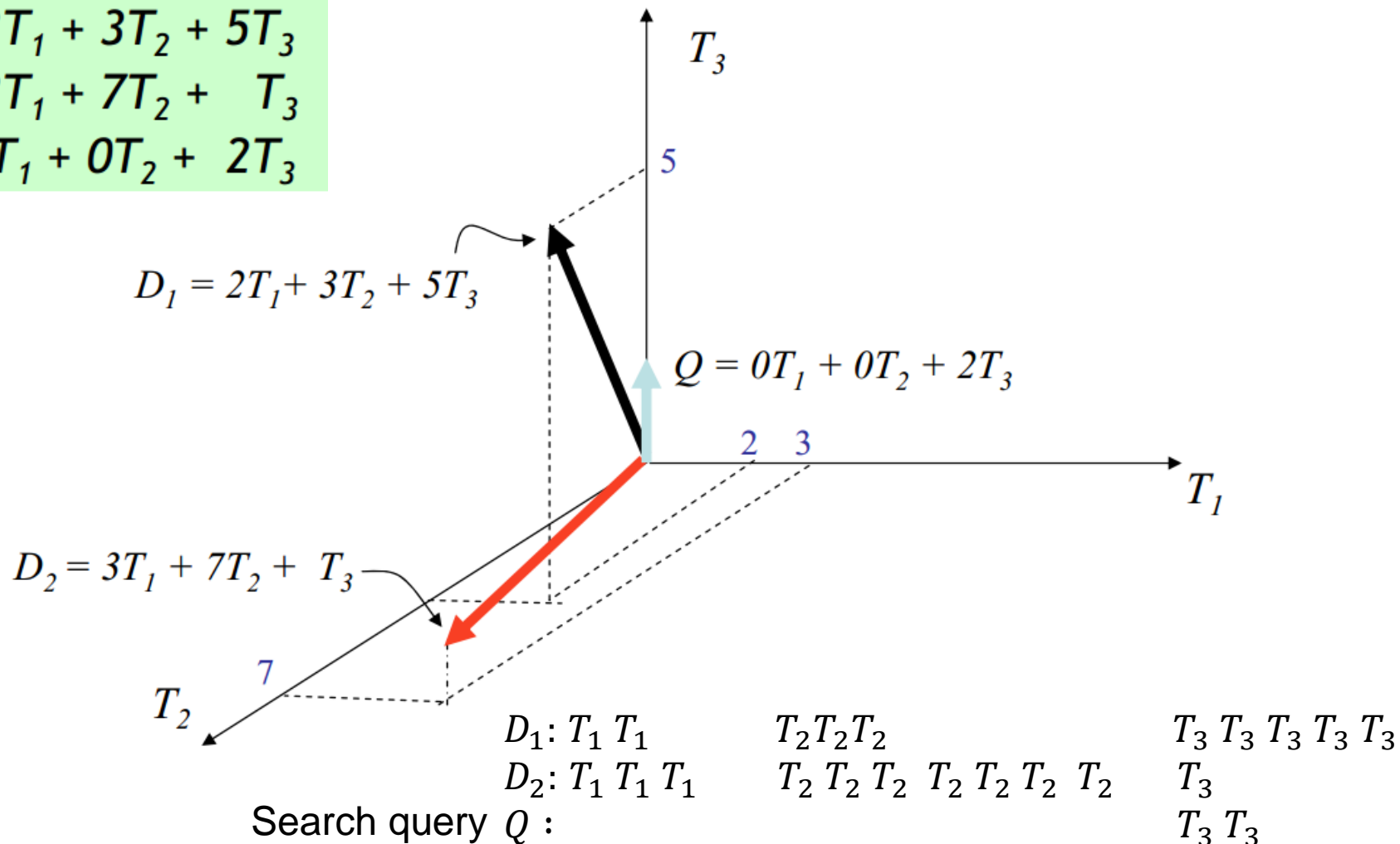
# Graphic Representation Vector Space Model

Example:

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

$$D_2 = 3T_1 + 7T_2 + T_3$$

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



# Graphic Representation Vector Space Model

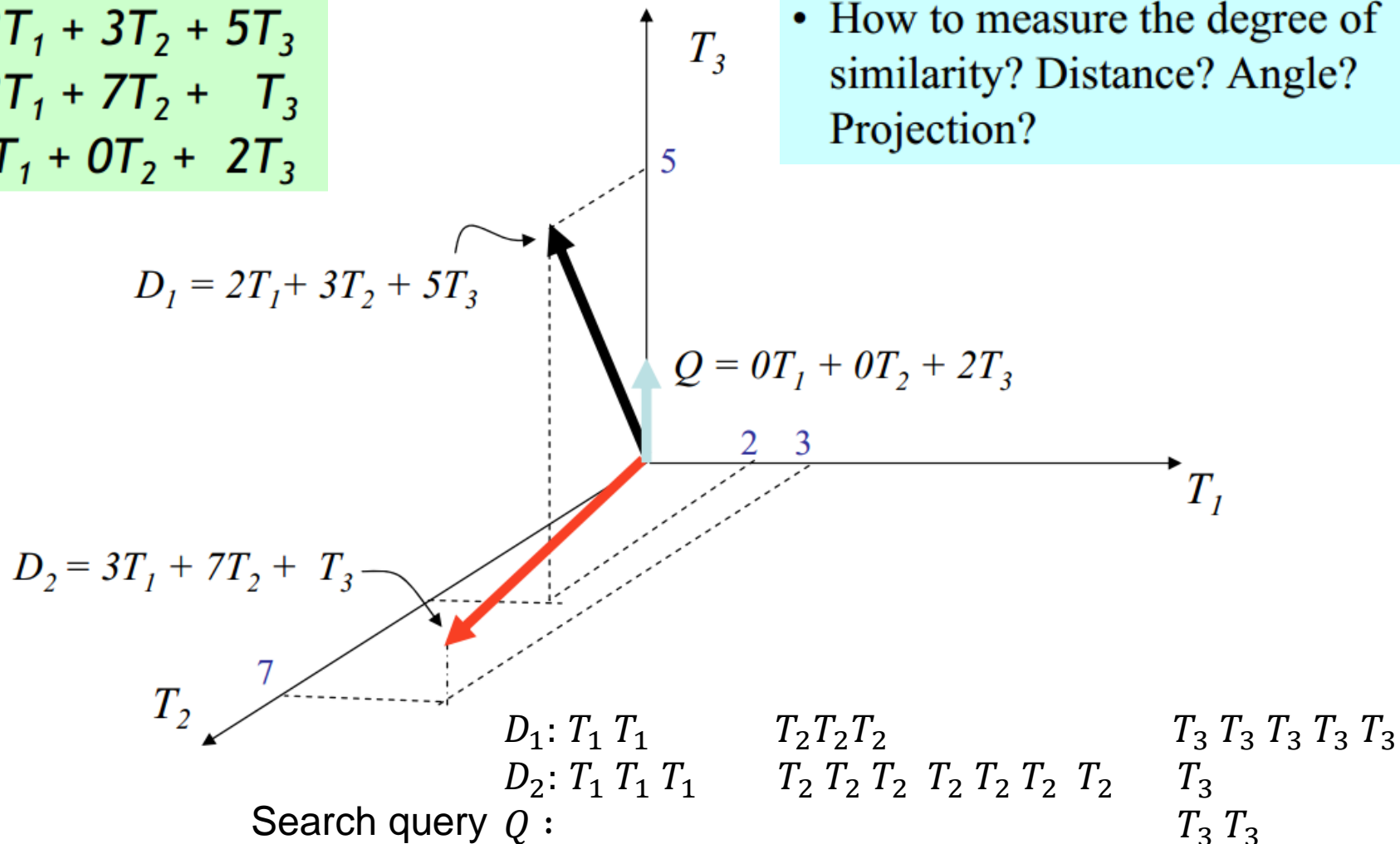
Example:

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

$$D_2 = 3T_1 + 7T_2 + T_3$$

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

- Is  $D_1$  or  $D_2$  more similar to  $Q$ ?
- How to measure the degree of similarity? Distance? Angle? Projection?



# How to measure degree of similarity?

## Use cosine similarity

$$sim(Q, D_i) = \frac{\sum_{j=1}^t w_{qj} * w_{d_{ij}}}{\sqrt{\sum_{j=1}^t (w_{qj})^2 * \sum_{j=1}^t (w_{d_{ij}})^2}}$$

# MATLAB Scripts

```
function [cosineSimilarity]=calculateCosineSimilarity(vectorX,vectorY)

    cosineSimilarity=sum(vectorX.*vectorY)/sqrt(sum(vectorX.^2)*sum(vectorY.^2));

end
```

$$sim(Q, D_i) = \frac{\sum_{j=1}^t w_{qj} * w_{d_{ij}}}{\sqrt{\sum_{j=1}^t (w_{qj})^2 * \sum_{j=1}^t (w_{d_{ij}})^2}}$$

# MATLAB Scripts

```
function [cosineSimilarity]=calculateCosineSimilarity(vectorX,vectorY)

    cosineSimilarity=sum(vectorX.*vectorY)/sqrt(sum(vectorX.^2)*sum(vectorY.^2));

end
```

```
>> D1=[2 3 5]
```

```
D1 =
     2     3     5
```

```
>> D2=[3 7 1]
```

```
D2 =
     3     7     1
```

```
>> Q=[0 0 2]
```

```
Q =
     0     0     2
```

```
>> calculateCosineSimilarity(Q,D1)
```

```
ans =
    0.8111
```

```
>> calculateCosineSimilarity(Q,D2)
```

```
ans =
    0.1302
```

$\cos(Q, D_1) = 0.8111$  //  $D_1$  is more similar to  $Q$   
 $\cos(Q, D_2) = 0.1302$

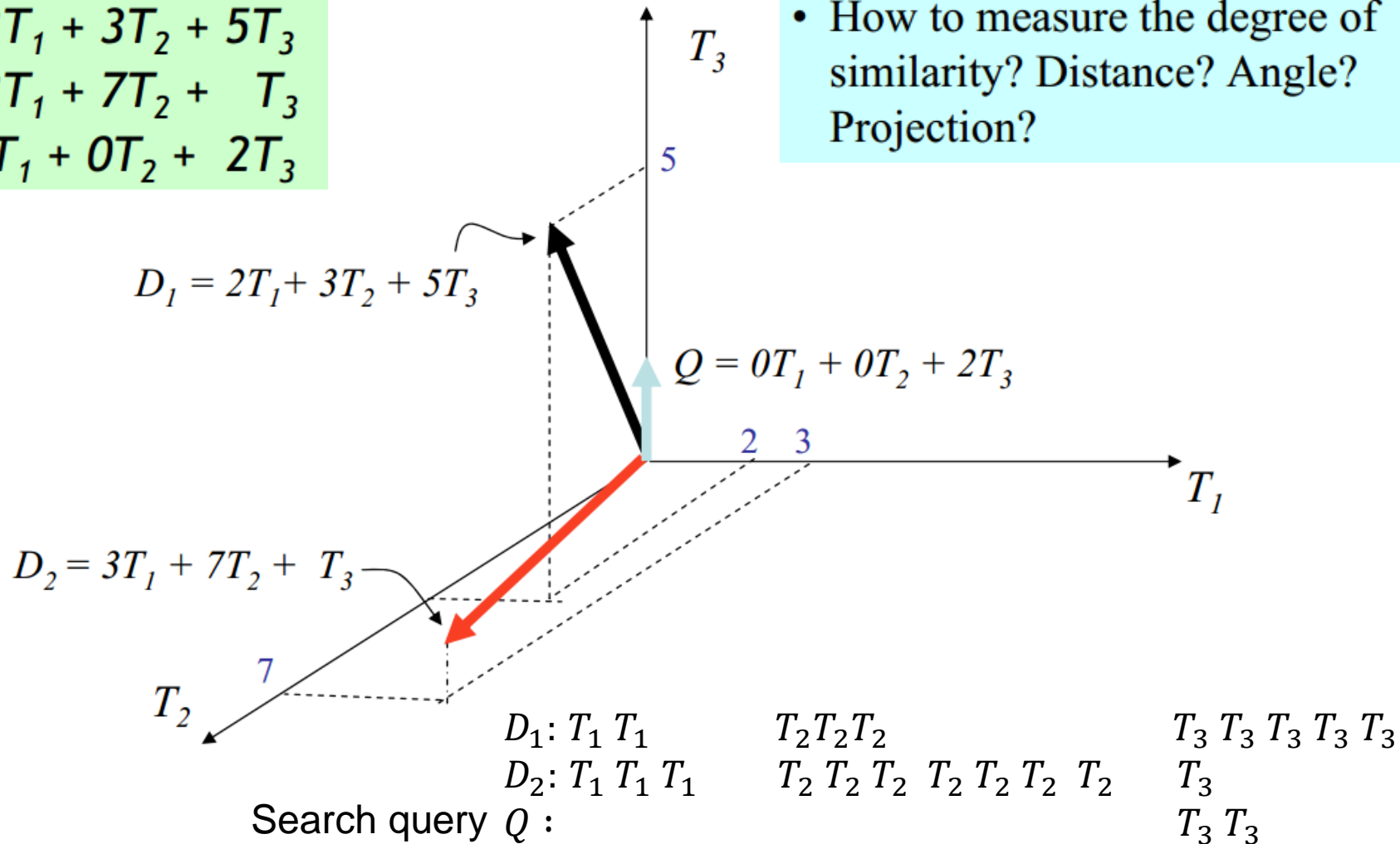
Example:

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

$$D_2 = 3T_1 + 7T_2 + T_3$$

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

- Is  $D_1$  or  $D_2$  more similar to  $Q$ ?
- How to measure the degree of similarity? Distance? Angle? Projection?

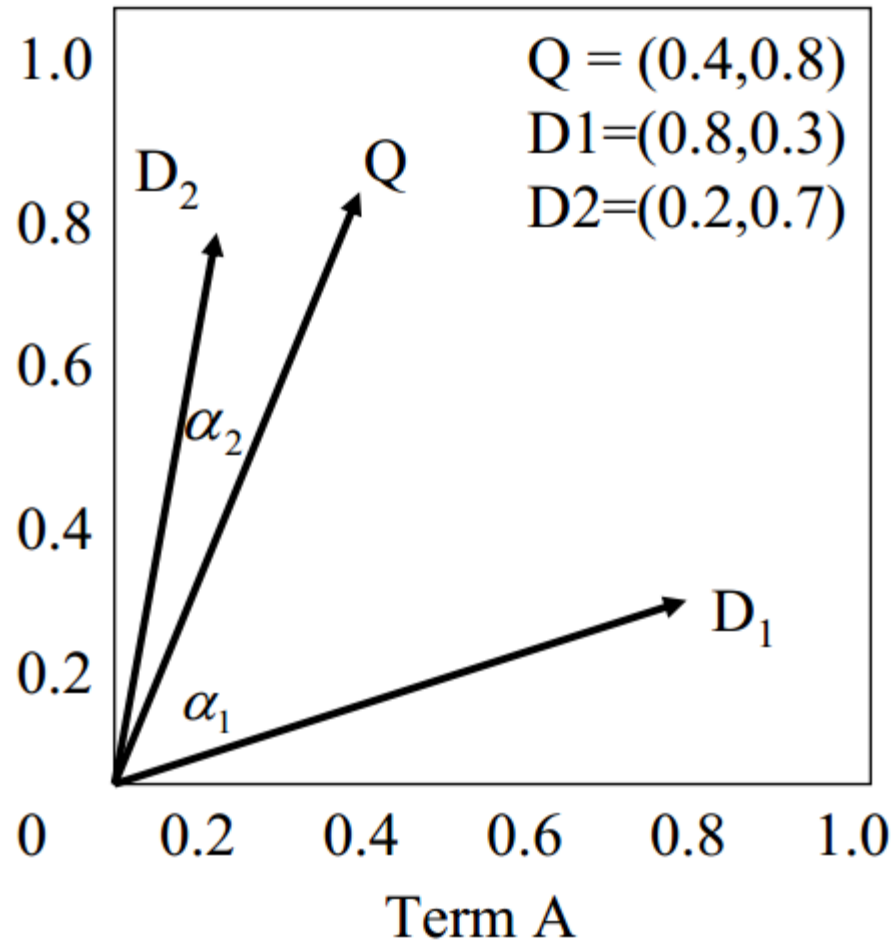


## Another VSM Model (2D)



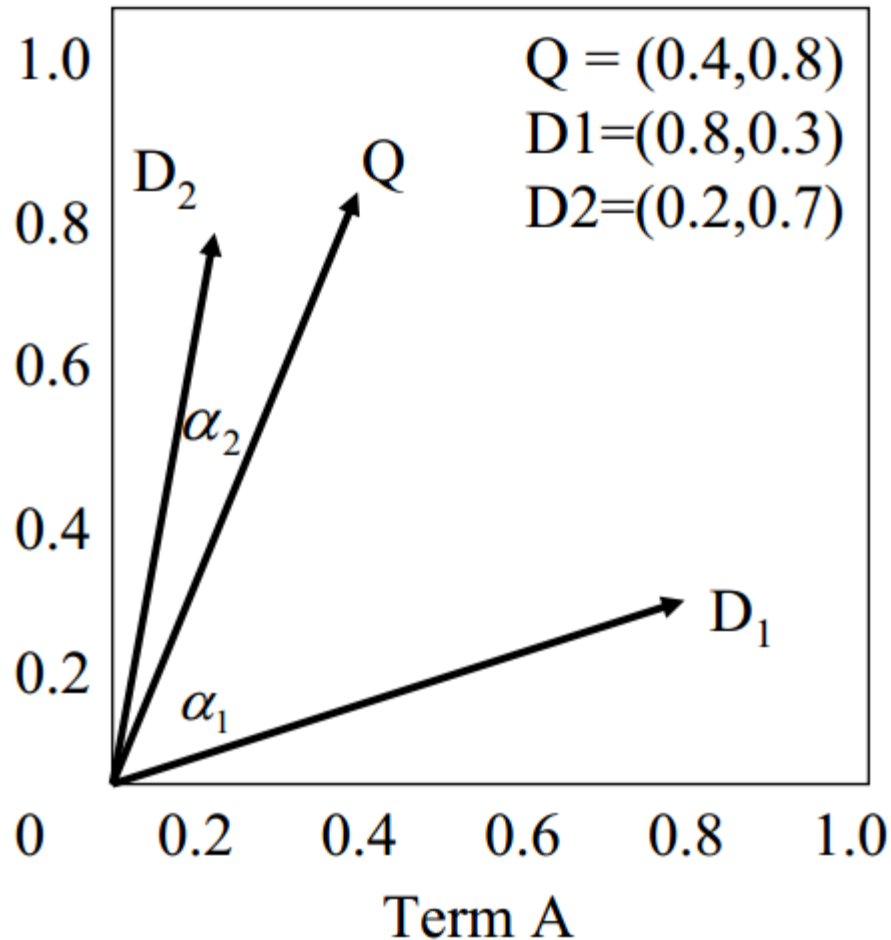
# Vector Space with Term Weights and Cosine Matching

Term B



# Vector Space with Term Weights and Cosine Matching

Term B



- Which document is most similar to  $Q$ ?
- Which document is most similar to  $Q$ ?

# Computing Relevance Scores

**Ex:** Query vector  $Q = (0.4, 0.8)$

Also, document  $D_2 = (0.2, 0.7)$

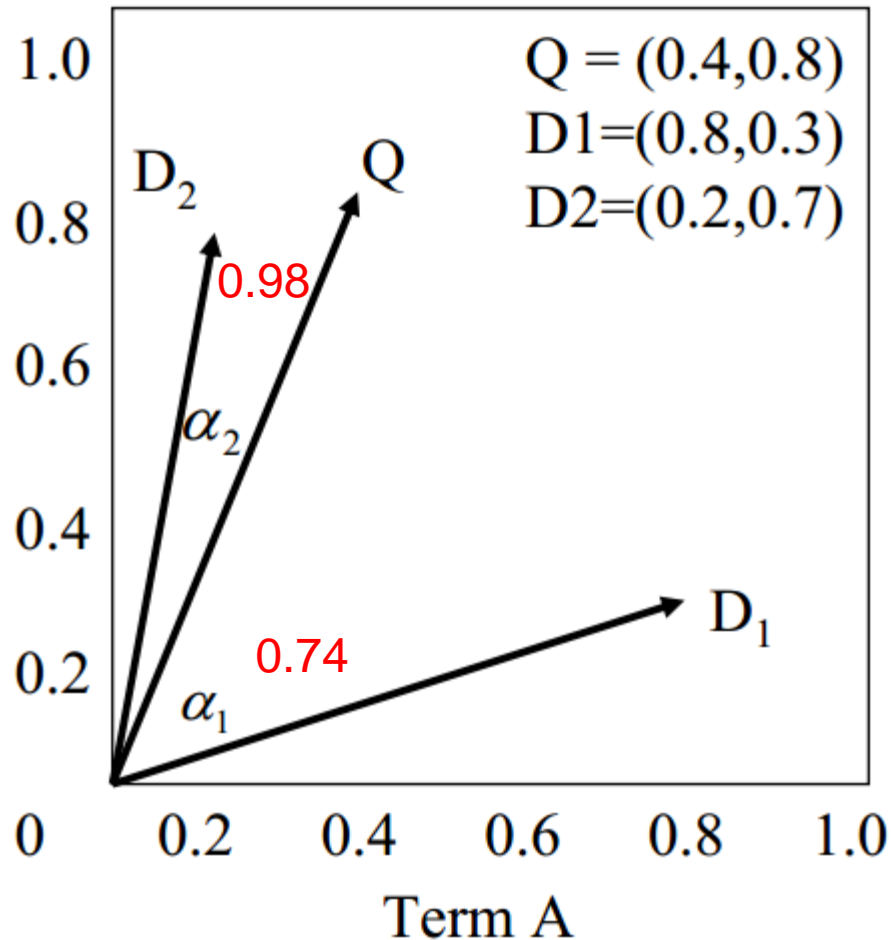
What does their similarity comparison yield?

$$sim(Q, D_i) = \frac{\sum_{j=1}^t w_{qj} * w_{d_{ij}}}{\sqrt{\sum_{j=1}^t (w_{qj})^2 * \sum_{j=1}^t (w_{d_{ij}})^2}}$$

$$\begin{aligned} sim(Q, D_2) &= \frac{(0.4 * 0.2) + (0.8 * 0.7)}{\sqrt{(0.4)^2 + (0.8)^2} * \sqrt{(0.2)^2 + (0.7)^2}} \\ &= \frac{0.64}{\sqrt{0.42}} = 0.98 \end{aligned}$$

# Vector Space with Term Weights and Cosine Matching

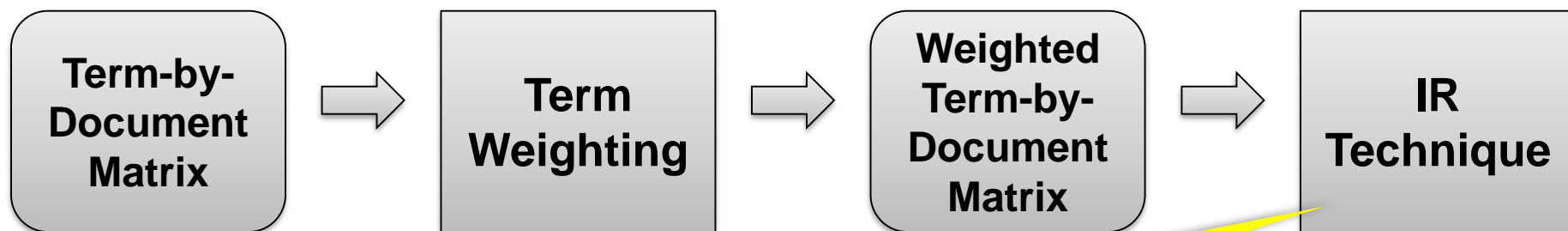
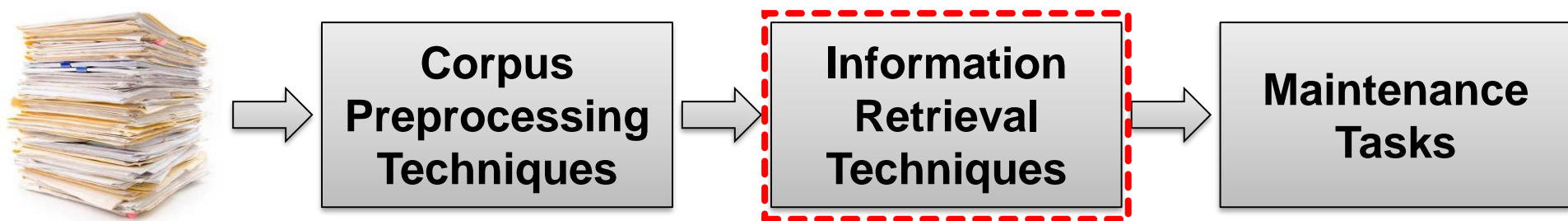
Term B



$$\text{sim}(Q, D_i) = \frac{\sum_{j=1}^t w_{q_j} w_{d_{ij}}}{\sqrt{\sum_{j=1}^t (w_{q_j})^2 \sum_{j=1}^t (w_{d_{ij}})^2}}$$

$$\begin{aligned} \text{sim}(Q, D2) &= \frac{(0.4 \cdot 0.2) + (0.8 \cdot 0.7)}{\sqrt{[(0.4)^2 + (0.8)^2] \cdot [(0.2)^2 + (0.7)^2]}} \\ &= \frac{0.64}{\sqrt{0.42}} = 0.98 \end{aligned}$$

$$\text{sim}(Q, D_1) = \frac{.56}{\sqrt{0.58}} = 0.74$$



**Vector Space Model**  
**Latent Semantic Indexing**  
**Latent Dirichlet Allocation**  
**Relational Topic Model**

# Why Latent Semantic Indexing (LSI)?

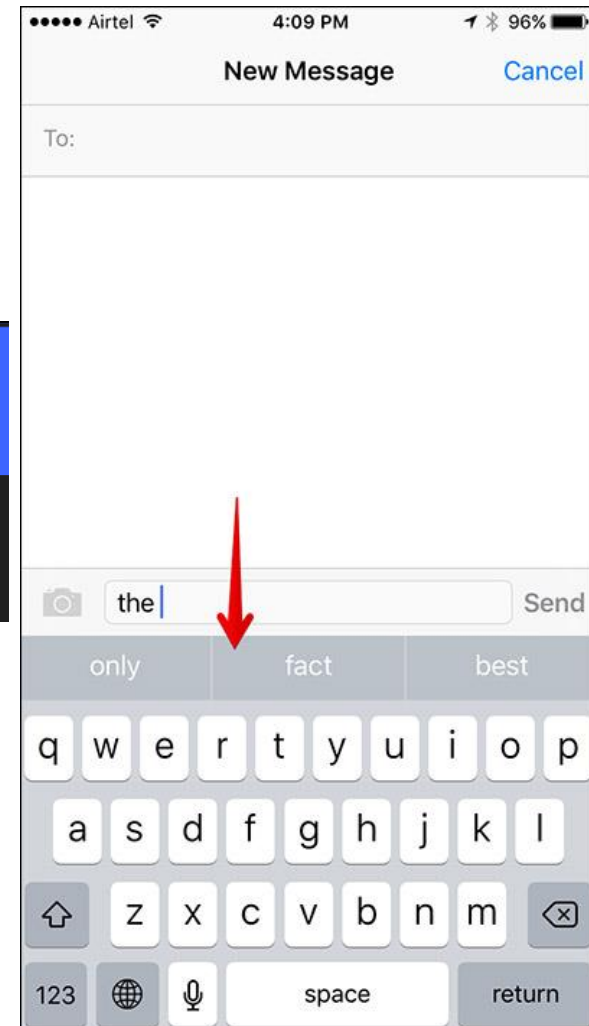
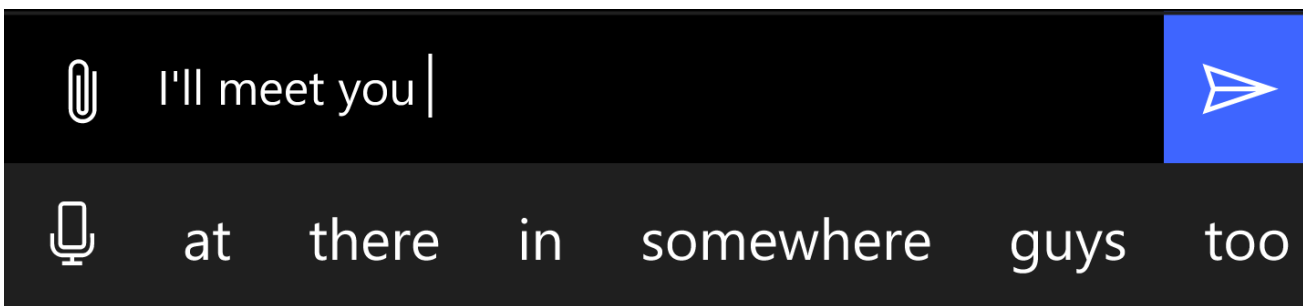
- Some problems for retrieval methods are based on term matching
  - vector-space similarity approach **works only if the terms of the query are explicitly present in the relevant documents**
- Rich expressive power of natural language
  - often queries contain terms that express ***concepts*** related to text to be retrieved

# Why Latent Semantic Indexing (LSI)?

- With the vector space model, we are **assuming independence among terms** in a document
  - ... however we know this is not true!!

# Why Latent Semantic Indexing (LSI)?

- With the vector space model, we are **assuming independence among terms** in a document
  - ... however we know this is not true!!





# Two Problems – Synonyms

- The same concept can be expressed using different sets of terms (**synonyms**), e.g.,:
  - True, right, correct
  - right place vs. correct position
  - window vs. frame
  - sort vs. order
- Negatively affects recall

# Two Problems – Homonyms

- Identical terms can be used in very different semantic contexts (**homonyms**), e.g.,:
  - bank (financial establishment vs. land near body of water)
  - chip
  - str, id, token
- Negatively affects precision

# LSI Idea

- Idea (Deerwester et al.):

“We would like a representation in which a set of terms, which by itself is incomplete and unreliable evidence of the relevance of a given document, is replaced by some other set of entities which are more reliable indicants. We take advantage of the implicit higher-order (or latent) structure in the association of terms and documents to reveal such relationships.”

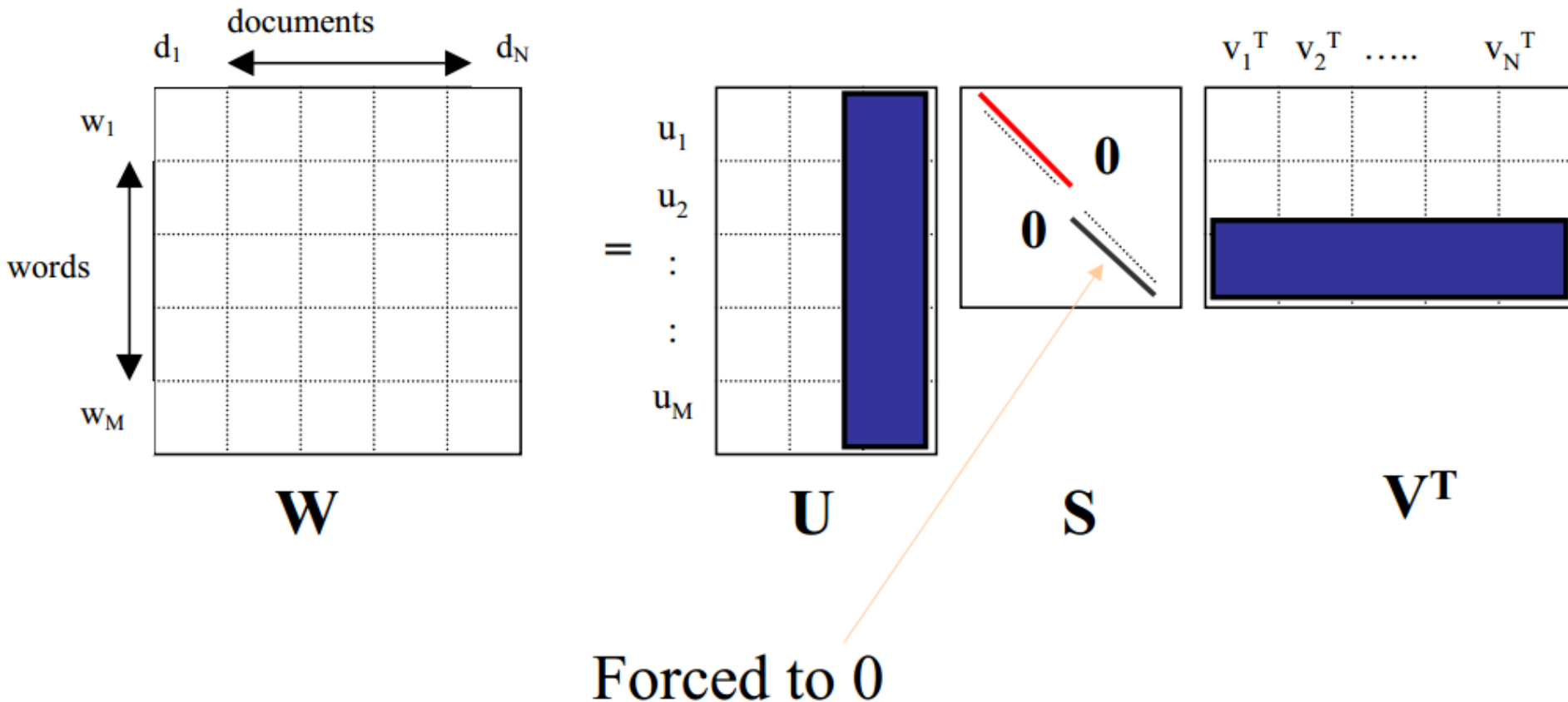
# Using SVD

- LSI uses linear algebra technique called **singular value decomposition (SVD)**
  - attempts to estimate the hidden structure
  - discovers the most important associative patterns between words and concepts
- In other words...
  - The analysis is moved from the space of terms to the space of concepts/topics

# Basically...

- Instead of representing documents as a set of correlated factors (**terms**), we represent documents as set of uncorrelated factors (**concepts**)

# SVD: Dimensionality Reduction



# LSI Example

■ Given a collection of documents (corpus):

- d1: Indian government goes for open-source software
- d2: Debian 3.0 Woody released
- d3: Wine 2.0 released with fixes for Gentoo 1.4 and Debian 3.0
- d4: gnuPOD released: iPOD on Linux... with GPLed software
- d5: Gentoo servers running at open-source mysql database
- d6: Dolly the sheep not totally identical clone
- d7: DNA news: introduced low-cost human genome DNA chip
- d8: Malaria-parasite genome database on the Web
- d9: UK sets up genome bank to protect rare sheep breeds
- d10: Dolly's DNA damaged

# LSI Example

- Given a collection of documents (corpus):

d1: Indian government goes for open-source software

d2: Debian 3.0 Woody released

d3: Wine 2.0 released with fixes for Gentoo 1.4 and Debian 3.0

d4: gnuPOD released: iPOD on Linux... with GPLed software

d5: Gentoo servers running at open-source mysql database

d6: Dolly the sheep not totally identical clone

d7: DNA news: introduced low-cost human genome DNA chip

d8: Malaria-parasite genome database on the Web

d9: UK sets up genome bank to protect rare sheep breeds

d10: Dolly's DNA damaged

d3 is about (open source) Linux versions...

d8 is about (DNA) manipulation...



# LSI Example:

## term-documents matrix

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	1	0	0	0	1	0	0	0	0	0
software	1	0	0	1	0	0	0	0	0	0
Linux	0	0	0	1	0	0	0	0	0	0
released	0	1	1	1	0	0	0	0	0	0
Debian	0	1	1	0	0	0	0	0	0	0
Gentoo	0	0	1	0	1	0	0	0	0	0
database	0	0	0	0	1	0	0	1	0	0
Dolly	0	0	0	0	0	1	0	0	0	1
sheep	0	0	0	0	0	1	0	0	0	0
genome	0	0	0	0	0	0	1	1	1	0
DNA	0	0	0	0	0	0	2	0	0	1

# LSI Example: term-documents matrix

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	1	0	0	0	1					
software	1	0	0	1	0					
Linux	0	0	0	1	0					
released	0	1	1	1	0					
Debian	0	1	1	0	0					
Gentoo	0	0	1	0	1					
database	0	0	0	0	1					
Dolly	0	0	0	0	0					
sheep	0	0	0	0	0					
genome	0	0	0	0	0					
DNA	0	0	0	0	0	0	2	0	0	1

d3 is about (open source) Linux versions, but...

# LSI Example: term-documents matrix

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	1	0	0	0	1					
software	1	0	0	1	0					
Linux	0	0	0	1	0					
released	0	1	1	1	0					
Debian	0	1	1	0	0					
Gentoo	0	0	1	0	1					
database	0	0	0	0	1					
Dolly	0	0	0	0	0					
sheep	0	0	0	0	0					
genome	0	0	0	0	0					
DNA	0	0	0	0	0	0	2	0	0	1

d3 is about (open source) Linux versions, but...

$\text{sim}(d1, d3) = 0$

$\text{sim}(d1, d5) = 0.7$

# LSI Example: term-documents matrix

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	1					0	0	0	0	0
software							0	0	0	0
Linux							0	0	0	0
released							0	0	0	0
Debian							0	0	0	0
Gentoo							0	0	0	0
database							0	1	0	0
Dolly							0	0	0	1
sheep							0	0	0	0
genome	0					0	1	1	1	0
DNA	0	0	0	0	0	0	2	0	0	1

d8 is about (DNA) manipulation, but...

# LSI Example: term-documents matrix

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	1					0	0	0	0	0
software							0	0	0	0
Linux							0	0	0	0
released							0	0	0	0
Debian							0	0	0	0
Gentoo							0	0	0	0
database							0	1	0	0
Dolly							0	0	0	1
sheep							0	0	0	0
genome	0					0	1	1	1	0
DNA	0	0	0	0	0	0	2	0	0	1

d8 is about (DNA) manipulation, but...

$\text{sim}(d8, d10)=0$   
 $\text{sim}(d7, d10)=0.63$

# Reconstructed Term-Document Matrix from the decomposed SVD matrices (k=2)

$$X' = U' * \Sigma' * V'^T$$

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	0.119	0.253	0.346	0.26	0.188	0.002	0.05	0.062	0.02	0.017
software	0.149	0.319	0.435	0.328	0.23	-0.01	-0.02	0.054	0.005	-0.02
Linux	0.102	0.22	0.299	0.226	0.158	-0	-0.02	0.035	0.002	-0.02
released	0.337	0.724	0.986	0.744	0.522	-0.01	-0.06	0.118	0.008	-0.05
Debian	0.235	0.505	0.687	0.519	0.364	-0.01	-0.04	0.083	0.006	-0.03
Gentoo	0.208	0.445	0.606	0.457	0.326	-0	0.028	0.092	0.021	0.002
database	0.092	0.187	0.259	0.193	0.159	0.025	0.258	0.111	0.067	0.117
Dolly	-0	-0.02	-0.02	-0.02	0.022	0.044	0.419	0.12	0.098	0.198
sheep	-0	-0	-0.01	-0	0.003	0.008	0.077	0.022	0.018	0.036
genome	0.025	0.014	0.034	0.014	0.107	0.116	1.107	0.329	0.262	0.521
DNA	0.002	-0.06	-0.06	-0.06	0.114	0.19	1.795	0.518	0.422	0.846

# Reconstructed Term-Document Matrix from the decomposed SVD matrices (k=2)

$$X' = U' * \Sigma' * V'^T$$

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10
open-source	0.119	0.253	0.346	0.26	0.188	0.002	0.05	0.062	0.02	0.017
software	0.149	0.319	0.435	0.328	0.23	-0.01	-0.02	0.054	0.005	-0.02
Linux	0.102	0.22	0.299	0.226	0.158	-0	-0.02	0.035	0.002	-0.02
released	0.337	0.724	0.986	0.744	0.522	-0.01	-0.06	0.118	0.008	-0.05
Debian	0.235									-0.03
Gentoo	0.208									0.002
database	0.092									0.117
Dolly	-0	-0.02	-0.02	-0.02	0.022	0.044	0.419	0.12	0.098	0.198
sheep	-0	-0	-0.01	-0	0.003	0.008	0.077	0.022	0.018	0.036
genome	0.025	0.014	0.034	0.014	0.107	0.116	1.107	0.329	0.262	0.521
DNA	0.002	-0.06	-0.06	-0.06	0.114	0.19	1.795	0.518	0.422	0.846

d3 is related to open-source...  
d8 is related to DNA...

# LSI: Pros and Cons

- + Able to deal with synonymy and homonymy
- + Stemming could be avoided (but works better with stemming!)
- + Increases similarity between documents of the same cluster
- + Decreases similarity between documents of different clusters
- More expensive than traditional Vector Space Models (SVD computation)
- Difficult to add new documents
- Determining the optimal  $k$  is a crucial issue
- Often needs a large document corpus



Concept Location

Impact Analysis



**Change Request**

**Concept Location**

**Impact Analysis**

**Implementation**

**Testing**

**Delivery**

# Incremental Change

[Rajlich'04]

Change Request

Concept Location

Impact Analysis

Implementation

Testing


Delivery

Bug 597338 - Performance Issue, hovering Bookmark menuitem highlight is lagged

**Status:** RESOLVED FIXED  
**Whiteboard:**  
**Keywords:** perf, regression  
**Product:** Firefox ([show info](#))  
**Component:** Location Bar ([show info](#))  
**Version:** Trunk  
**Platform:** All All  
**Reported:** 2010-09-17 00:21 PDT by Alice0775 White  
**Modified:** 2010-10-30 03:37 PDT ([History](#))  
**CC List:** 14 users ([show](#))  
**See Also:**  
**Crash Signature:**  
**Project Flags:**

#### Attachments

[patch 1.1 with test fix](#) (8.62 KB, patch) 2010-09-20 15:46 PDT, Drew Willcoxon, jadv, no flags, Details | Diff  
[patch: don't get computed style on each call](#) (11.47 KB, patch) 2010-09-20 17:19 PDT, Drew Willcoxon, jadv, dao: review+, Details | Diff  
[Add an attachment](#) (proposed patch, testcase, etc.) Show

 **Alice0775 White** 2010-09-17 00:21:25 PDT

Build Identifier: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0b7pre) Gecko/20100916 Firefox/3.6.0  
ID:20100916041016

In case of Long Bookmarks list,  
hovering Bookmarks/history menuitem highlight is lagged.

This problem is more remarkable in Windows 7 Classic and/or old machine.

Reproducible: Always

#### Steps to Reproduce:

1. Start Minefield with new profile
2. Make 100 bookmarks in a folder
3. Open the folder. (or "Latest Headlines" on Bookmarks Toolbar).
4. Hover and move mouse pointer on menu item up and down

## Bug Report Task



## Concept Location

# Implementation

## Delivery

Bug 597338 - Performance Issue, hovering Bookmark menuitem highlight is lagged

Status: RESOLVED FIXED

Reported: 2010-09-17 00:21 PDT by Alice0775 White

Whiteboard:

Modified: 2010-10-30 03:37 PDT ([History](#))

Keywords: perf, regression

CC List: 14 users ([show](#))

Product: Firefox ([show info](#))

See Also:

Component: Location Bar ([show info](#))

Version: Trunk

Platform: All All

Crash Signature:

Project Flags:

Attachments

patch 1.1 with test fix (8.62 KB, patch)

2010-09-20 15:46 Patch, Drew Wilkinson >add>

no flags

Details |

patch: don't get computed style on each call (11.47 KB, patch)

2010-09-20 17:19 Patch, Drew Wilkinson >add>

dao: review+

Details |

Add an attachment (proposed patch, testcase, etc.)

Sho

Alice0775 White

2010-09-17 00:21:25 PDT

Build Identifier: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0b7pre) Gecko/20100916 Firefox/20100916041016

In case of Long Bookmarks list, hovering Bookmark/history menuitem highlight is lagged.

This problem is more remarkable in Windows 7 Classic and/or old machine.

Reproducible: Always

Steps to Reproduce:

1. Start Hinesfield with new profile

2. Make 100 bookmarks in a folder

3. Open the folder (or "Latest Headlines" on Bookmarks Toolbar).

4. Hover and move mouse pointer on menu item up and down

**Initial Code Snippet**

```

def find_max(arr):
    max_val = arr[0]
    for i in range(1, len(arr)):
        if arr[i] > max_val:
            max_val = arr[i]
    return max_val

```

**Evolution 1: Adding a 'max' variable**

```

def find_max(arr):
    max_val = arr[0]
    for i in range(1, len(arr)):
        if arr[i] > max_val:
            max_val = arr[i]
    return max_val

```

**Evolution 2: Adding a 'maxIndex' variable**

```

def find_max(arr):
    max_val = arr[0]
    max_index = 0
    for i in range(1, len(arr)):
        if arr[i] > max_val:
            max_val = arr[i]
            max_index = i
    return max_val, max_index

```

**Evolution 3: Adding a 'max' variable and a 'maxIndex' variable**

```

def find_max(arr):
    max_val = arr[0]
    max_index = 0
    for i in range(1, len(arr)):
        if arr[i] > max_val:
            max_val = arr[i]
            max_index = i
    return max_val, max_index

```

**Evolution 4: Adding a 'max' variable and a 'maxIndex' variable**

```

def find_max(arr):
    max_val = arr[0]
    max_index = 0
    for i in range(1, len(arr)):
        if arr[i] > max_val:
            max_val = arr[i]
            max_index = i
    return max_val, max_index

```

**Legend:**

- Initial Code Snippet
- Evolution 1: Adding a 'max' variable
- Evolution 2: Adding a 'maxIndex' variable
- Evolution 3: Adding a 'max' variable and a 'maxIndex' variable
- Evolution 4: Adding a 'max' variable and a 'maxIndex' variable

## Change Request

## Concept Location

## Impact Analysis

## Implementation

## Testing

## Delivery

## Source Code

Bug 597338 - Performance Issue, hovering Bookmark menuitem highlight is lagged

<b>Status:</b>	RESOLVED FIXED	<b>Reported:</b>	2010-09-17 00:21 PDT by Alico0775 White
<b>Whiteboard:</b>		<b>Modified:</b>	2010-10-30 03:37 PDT ( <a href="#">History</a> )
<b>Keywords:</b>	perf, regression	<b>CC List:</b>	14 users ( <a href="#">show</a> )
<b>Product:</b>	Firefox ( <a href="#">show info</a> )	<b>See Also:</b>	
<b>Component:</b>	Location Bar ( <a href="#">show info</a> )		
<b>Version:</b>	Trunk	<b>Crash Signature:</b>	
<b>Platform:</b>	All All	<b>Project Files:</b>	

Attachments		
<p><b>patch 1.1 with test fix</b> (8.62 KB, patch)</p> <p>2010-09-20 15:46 PDT, Drew Willcoxon &lt;advice@mozilla.org&gt;</p>	no flags	<a href="#">Details</a>
<p><b>patch: don't get computed style on each call</b> (11.47 KB, patch)</p> <p>2010-09-20 17:19 PDT, Drew Willcoxon &lt;advice@mozilla.org&gt;</p>	dao: review+	<a href="#">Details</a>
<p><a href="#">Add an attachment</a> (proposed patch, testcase, etc.)</p>		<a href="#">Show all</a>

 **Alice0775 White** 2010-09-17 00:21:25 PDT

Build Identifier: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0b7pre) Gecko/20100916 Firefox/3.6.10  
ID:20100916041016

In case of Long Bookmarks list,  
hovering Bookmarks/history menuitem highlight is lagged.

This problem is more remarkable in Windows 7 Classic and/or old machine.

Reproducible: Always

Steps to Reproduce:

1. Start Minefield with new profile
2. Make 100 bookmarks in a folder
3. Open the folder (or "Latest Headlines" on Bookmarks Toolbar).
4. Hover and move mouse pointer on menu item up and down

[illegible]

```

Function CheckIsPrime(Val As Integer) As Boolean
Dim i As Integer
Dim IsPrime As Boolean

If Val < 2 Then
    IsPrime = False
Else
    For i = 2 To Val - 1
        If Val Mod i = 0 Then
            IsPrime = False
            Exit For
        End If
    Next i

    If IsPrime Then
        IsPrime = True
    End If
End If

Return IsPrime
End Function

Function IsPrime(Val As Integer) As Boolean
Dim i As Integer
Dim IsPrime As Boolean

If Val < 2 Then
    IsPrime = False
Else
    For i = 2 To Val - 1
        If Val Mod i = 0 Then
            IsPrime = False
            Exit For
        End If
    Next i

    If IsPrime Then
        IsPrime = True
    End If
End If

Return IsPrime
End Function

```

[illegible][illegible]

```

Open Function CheckIfValid (id As Integer) As Boolean
    Dim isValid As Boolean
    Dim idStr As String
    Dim i As Integer
    Dim ch As Character
    Dim str As String

    'Starts with 0 or 1
    idStr = CStr(id)
    If Left(idStr, 1) = "0" Or Left(idStr, 1) = "1" Then
        isValid = False
        Return False
    End If

    'Starts with 00 or 01 as a comment
    If Left(idStr, 2) = "00" Or Left(idStr, 2) = "01" Then
        isValid = False
        Return False
    End If

    'Check if a number and if it is a comment or not
    For i = 2 To Len(idStr)
        ch = Mid(idStr, i, 1)
        If Not (ch <= "9" Or ch = ".") Then
            'Not a number
            isValid = False
            Return False
        End If
    Next i

    'If no float = " " then an even number of " " characters
    'If no float = " " then an odd number of " " characters means
    'that it is the start of a comment, and otherwise means
    'that it is a number
    If Not (idStr = " ") Then
        If (Len(idStr) Mod 2) = 0 Then
            isValid = True
        Else
            isValid = False
        End If
    Else
        'Starts with " "
        isValid = True
    End If

    CheckIsValid = isValid
End Function
End Sub

'CheckIsValid = isValid
End Sub

```

[illegible]

```

Petersen P = Graph(10, [[0, 1], [0, 6], [1, 2], [1, 7], [2, 3], [2, 8], [3, 4], [3, 9], [4, 5], [4, 6], [5, 7], [5, 8], [6, 9], [7, 10], [8, 10], [9, 10]])
In [10]: P
Out[10]: Petersen graph
In [11]: P.degree(0)
Out[11]: 2
In [12]: P.degree(1)
Out[12]: 2
In [13]: P.degree(2)
Out[13]: 2
In [14]: P.degree(3)
Out[14]: 2
In [15]: P.degree(4)
Out[15]: 2
In [16]: P.degree(5)
Out[16]: 2
In [17]: P.degree(6)
Out[17]: 2
In [18]: P.degree(7)
Out[18]: 2
In [19]: P.degree(8)
Out[19]: 2
In [20]: P.degree(9)
Out[20]: 2
In [21]: P.degree(10)
Out[21]: 2
In [22]: P.degree(11)
Out[22]: 0
In [23]: P.degree(12)
Out[23]: 0
In [24]: P.degree(13)
Out[24]: 0
In [25]: P.degree(14)
Out[25]: 0
In [26]: P.degree(15)
Out[26]: 0
In [27]: P.degree(16)
Out[27]: 0
In [28]: P.degree(17)
Out[28]: 0
In [29]: P.degree(18)
Out[29]: 0
In [30]: P.degree(19)
Out[30]: 0
In [31]: P.degree(20)
Out[31]: 0
In [32]: P.degree(21)
Out[32]: 0
In [33]: P.degree(22)
Out[33]: 0
In [34]: P.degree(23)
Out[34]: 0
In [35]: P.degree(24)
Out[35]: 0
In [36]: P.degree(25)
Out[36]: 0
In [37]: P.degree(26)
Out[37]: 0
In [38]: P.degree(27)
Out[38]: 0
In [39]: P.degree(28)
Out[39]: 0
In [40]: P.degree(29)
Out[40]: 0
In [41]: P.degree(30)
Out[41]: 0
In [42]: P.degree(31)
Out[42]: 0
In [43]: P.degree(32)
Out[43]: 0
In [44]: P.degree(33)
Out[44]: 0
In [45]: P.degree(34)
Out[45]: 0
In [46]: P.degree(35)
Out[46]: 0
In [47]: P.degree(36)
Out[47]: 0
In [48]: P.degree(37)
Out[48]: 0
In [49]: P.degree(38)
Out[49]: 0
In [50]: P.degree(39)
Out[50]: 0
In [51]: P.degree(40)
Out[51]: 0
In [52]: P.degree(41)
Out[52]: 0
In [53]: P.degree(42)
Out[53]: 0
In [54]: P.degree(43)
Out[54]: 0
In [55]: P.degree(44)
Out[55]: 0
In [56]: P.degree(45)
Out[56]: 0
In [57]: P.degree(46)
Out[57]: 0
In [58]: P.degree(47)
Out[58]: 0
In [59]: P.degree(48)
Out[59]: 0
In [60]: P.degree(49)
Out[60]: 0
In [61]: P.degree(50)
Out[61]: 0
In [62]: P.degree(51)
Out[62]: 0
In [63]: P.degree(52)
Out[63]: 0
In [64]: P.degree(53)
Out[64]: 0
In [65]: P.degree(54)
Out[65]: 0
In [66]: P.degree(55)
Out[66]: 0
In [67]: P.degree(56)
Out[67]: 0
In [68]: P.degree(57)
Out[68]: 0
In [69]: P.degree(58)
Out[69]: 0
In [70]: P.degree(59)
Out[70]: 0
In [71]: P.degree(60)
Out[71]: 0
In [72]: P.degree(61)
Out[72]: 0
In [73]: P.degree(62)
Out[73]: 0
In [74]: P.degree(63)
Out[74]: 0
In [75]: P.degree(64)
Out[75]: 0
In [76]: P.degree(65)
Out[76]: 0
In [77]: P.degree(66)
Out[77]: 0
In [78]: P.degree(67)
Out[78]: 0
In [79]: P.degree(68)
Out[79]: 0
In [80]: P.degree(69)
Out[80]: 0
In [81]: P.degree(70)
Out[81]: 0
In [82]: P.degree(71)
Out[82]: 0
In [83]: P.degree(72)
Out[83]: 0
In [84]: P.degree(73)
Out[84]: 0
In [85]: P.degree(74)
Out[85]: 0
In [86]: P.degree(75)
Out[86]: 0
In [87]: P.degree(76)
Out[87]: 0
In [88]: P.degree(77)
Out[88]: 0
In [89]: P.degree(78)
Out[89]: 0
In [90]: P.degree(79)
Out[90]: 0
In [91]: P.degree(80)
Out[91]: 0
In [92]: P.degree(81)
Out[92]: 0
In [93]: P.degree(82)
Out[93]: 0
In [94]: P.degree(83)
Out[94]: 0
In [95]: P.degree(84)
Out[95]: 0
In [96]: P.degree(85)
Out[96]: 0
In [97]: P.degree(86)
Out[97]: 0
In [98]: P.degree(87)
Out[98]: 0
In [99]: P.degree(88)
Out[99]: 0
In [100]: P.degree(89)
Out[100]: 0
In [101]: P.degree(90)
Out[101]: 0
In [102]: P.degree(91)
Out[102]: 0
In [103]: P.degree(92)
Out[103]: 0
In [104]: P.degree(93)
Out[104]: 0
In [105]: P.degree(94)
Out[105]: 0
In [106]: P.degree(95)
Out[106]: 0
In [107]: P.degree(96)
Out[107]: 0
In [108]: P.degree(97)
Out[108]: 0
In [109]: P.degree(98)
Out[109]: 0
In [110]: P.degree(99)
Out[110]: 0
In [111]: P.degree(100)
Out[111]: 0
In [112]: P.degree(101)
Out[112]: 0
In [113]: P.degree(102)
Out[113]: 0
In [114]: P.degree(103)
Out[114]: 0
In [115]: P.degree(104)
Out[115]: 0
In [116]: P.degree(105)
Out[116]: 0
In [117]: P.degree(106)
Out[117]: 0
In [118]: P.degree(107)
Out[118]: 0
In [119]: P.degree(108)
Out[119]: 0
In [120]: P.degree(109)
Out[120]: 0
In [121]: P.degree(110)
Out[121]: 0
In [122]: P.degree(111)
Out[122]: 0
In [123]: P.degree(112)
Out[123]: 0
In [124]: P.degree(113)
Out[124]: 0
In [125]: P.degree(114)
Out[125]: 0
In [126]: P.degree(115)
Out[126]: 0
In [127]: P.degree(116)
Out[127]: 0
In [128]: P.degree(117)
Out[128]: 0
In [129]: P.degree(118)
Out[129]: 0
In [130]: P.degree(119)
Out[130]: 0
In [131]: P.degree(120)
Out[131]: 0
In [132]: P.degree(121)
Out[132]: 0
In [133]: P.degree(122)
Out[133]: 0
In [134]: P.degree(123)
Out[134]: 0
In [135]: P.degree(124)
Out[135]: 0
In [136]: P.degree(125)
Out[136]: 0
In [137]: P.degree(126)
Out[137]: 0
In [138]: P.degree(127)
Out[138]: 0
In [139]: P.degree(128)
Out[139]: 0
In [140]: P.degree(129)
Out[140]: 0
In [141]: P.degree(130)
Out[141]: 0
In [142]: P.degree(131)
Out[142]: 0
In [143]: P.degree(132)
Out[143]: 0
In [144]: P.degree(133)
Out[144]: 0
In [145]: P.degree(134)
Out[145]: 0
In [146]: P.degree(135)
Out[146]: 0
In [147]: P.degree(136)
Out[147]: 0
In [148]: P.degree(137)
Out[148]: 0
In [149]: P.degree(138)
Out[149]: 0
In [150]: P.degree(139)
Out[150]: 0
In [151]: P.degree(140)
Out[151]: 0
In [152]: P.degree(141)
Out[152]: 0
In [153]: P.degree(142)
Out[153]: 0
In [154]: P.degree(143)
Out[154]: 0
In [155]: P.degree(144)
Out[155]: 0
In [156]: P.degree(145)
Out[156]: 0
In [157]: P.degree(146)
Out[157]: 0
In [158]: P.degree(147)
Out[158]: 0
In [159]: P.degree(148)
Out[159]: 0
In [160]: P.degree(149)
Out[160]: 0
In [161]: P.degree(150)
Out[161]: 0
In [162]: P.degree(151)
Out[162]: 0
In [163]: P.degree(152)
Out[163]: 0
In [164]: P.degree(153)
Out[164]: 0
In [165]: P.degree(154)
Out[165]: 0
In [166]: P.degree(155)
Out[166]: 0
In [167]: P.degree(156)
Out[167]: 0
In [168]: P.degree(157)
Out[168]: 0
In [169]: P.degree(158)
Out[169]: 0
In [170]: P.degree(159)
Out[170]: 0
In [171]: P.degree(160)
Out[171]: 0
In [172]: P.degree(161)
Out[172]: 0
In [173]: P.degree(162)
Out[173]: 0
In [174]: P.degree(163)
Out[174]: 0
In [175]: P.degree(164)
Out[175]: 0
In [176]: P.degree(165)
Out[176]: 0
In [177]: P.degree(166)
Out[177]: 0
In [178]: P.degree(167)
Out[178]: 0
In [179]: P.degree(168)
Out[179]: 0
In [180]: P.degree(169)
Out[180]: 0
In [181]: P.degree(170)
Out[181]: 0
In [182]: P.degree(171)
Out[182]: 0
In [183]: P.degree(172)
Out[183]: 0
In [184]: P.degree(173)
Out[184]: 0
In [185]: P.degree(174)
Out[185]: 0
In [186]: P.degree(175)
Out[186]: 0
In [187]: P.degree(176)
Out[187]: 0
In [188]: P.degree(177)
Out[188]: 0
In [189]: P.degree(178)
Out[189]: 0
In [190]: P.degree(179)
Out[190]: 0
In [191]: P.degree(180)
Out[191]: 0
In [192]: P.degree(181)
Out[192]: 0
In [193]: P.degree(182)
Out[193]: 0
In [194]: P.degree(183)
Out[194]: 0
In [195]: P.degree(184)
Out[195]: 0

```

```

Python Functions and Lists #11: add for strings to strings

def add(s1, s2):
    """
    Return the concatenation of two strings.

    Parameters:
    s1 (str): The first string.
    s2 (str): The second string.

    Returns:
    str: The concatenated string.
    """
    # Concatenate the strings
    result = s1 + s2

    # Return the result
    return result

# Test the function
s1 = "Hello"
s2 = "World"
result = add(s1, s2)

# Print the result
print(result)

# Expected output:
# HelloWorld

```

```
# Create a new DataFrame object from scratch
df = pd.DataFrame()

# Add columns
df['name'] = ['John', 'Doe', 'Jane', 'Bob', 'Alice']
df['age'] = [25, 30, 35, 40, 45]
df['gender'] = ['M', 'F', 'F', 'M', 'F']

# Add rows
df.loc[6] = ('Tom', 28, 'M')
df.loc[7] = ('Liam', 32, 'M')
df.loc[8] = ('Olivia', 29, 'F')
df.loc[9] = ('Noah', 38, 'M')
df.loc[10] = ('Ava', 33, 'F')
```

[illegible]

# Bug Report Task



Change Request

Concept Location

Impact Analysis

Implementation

Testing

Delivery

Source  
Code

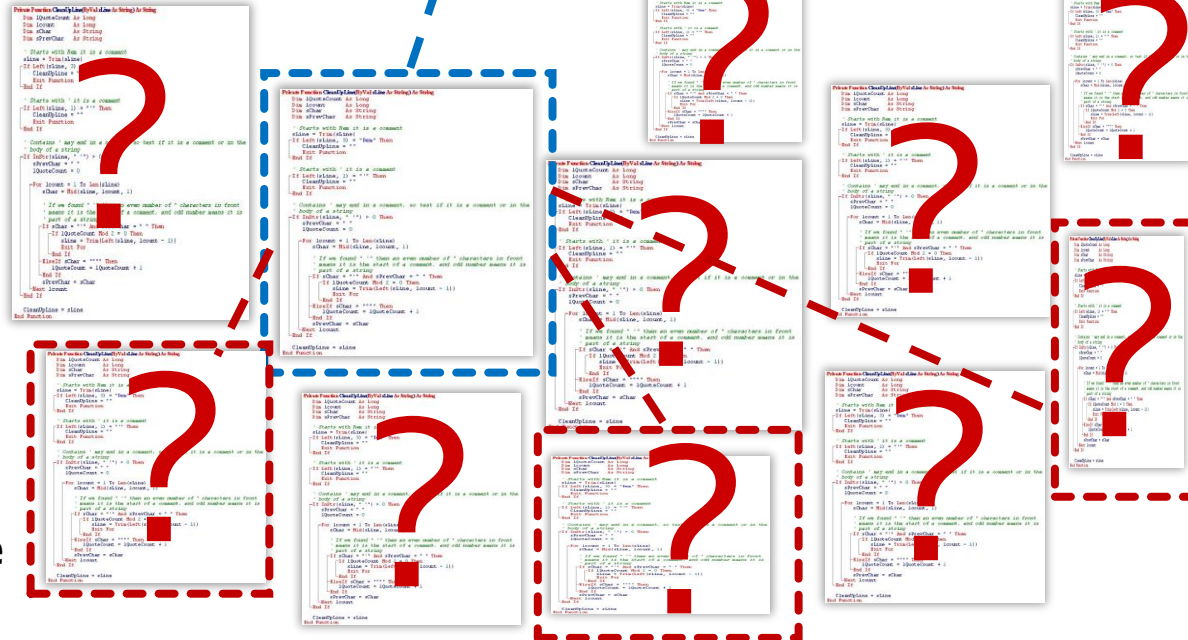
Bug 597338 - Performance Issue, hovering Bookmark menuitem highlight is lagged

Status: RESOLVED FIXED  
Whiteboard:  
Keywords: perf, regression  
Product: Firefox (show info)  
Component: Location Bar (show info)  
Version: Trunk  
Platform: All All  
Reported: 2010-09-17 00:21 PDT by Alice0775 White  
Modified: 2010-10-30 03:37 PDT (History)  
CC List: 14 users (show)  
See Also:  
Crash Signature:  
Project Flags:

Attachments  
patch 1.1 with test fix (8.63 KB, patch) no flags Details | Download  
2010-09-20 15:46 PDT: Drew Wilkinson <adw...@mozilla.org>  
patch: don't get computed style on each call (11.47 KB, patch) dao: review+ Details | Download  
2010-09-20 17:19 PDT: Drew Wilkinson <adw...@mozilla.org>  
Add an attachment (proposed patch, testcase, etc.) Show

Alice0775 White 2010-09-17 00:21:25 PDT  
Build Identifier: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0b7pre) Gecko/20100916 Firefox/3.6.10  
In case of Long Bookmarks list, hovering Bookmarks/history menuitem highlight is lagged.  
This problem is more remarkable in Windows 7 Classic and/or old machine.  
Reproducible: Always

Steps to Reproduce:  
1. Start Minefield with new profile  
2. Make 100 bookmarks in a folder  
3. Open the folder. (or "Latest Headlines" on Bookmarks Toolbar).  
4. Hover and move mouse pointer on menu item up and down



Change Request

Concept Location

Impact Analysis

Implementation

Testing

Delivery

Source  
Code

## Extremely challenging tasks:

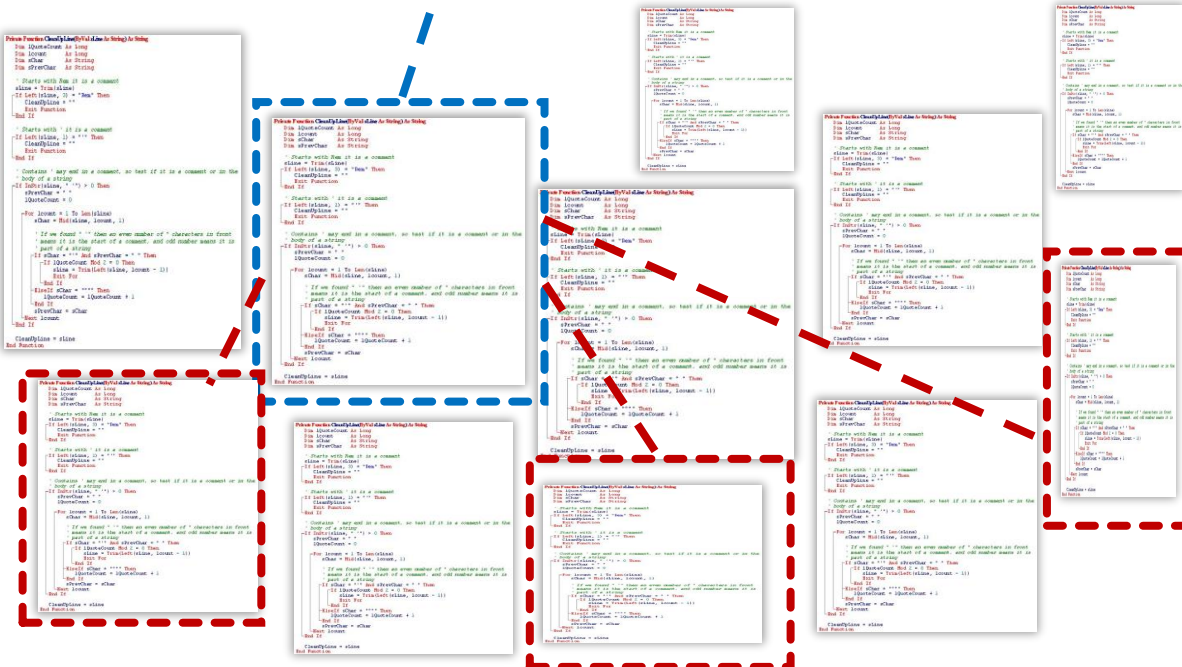
- Millions of Lines of Code
- Absence of original developer
- Missing or outdated documentation
- etc.

Bug 597338 - Performance Issue, hovering Bookmark menuitem highlight is lagged

Status: RESOLVED FIXED  
Whiteboard:  
Keywords: perf, regression  
Product: Firefox (show info)  
Component: Location Bar (show info)  
Version: Trunk  
Platform: All All  
Reported: 2010-09-17 00:21 PDT by Alice0775 White  
Modified: 2010-10-30 03:37 PDT (History)  
CC List: 14 users (show)  
See Also:  
Crash Signature:  
Project Flags:

Attachments  
patch 1.1 with test fix (8.62 KB, patch)  
2010-09-20 15:46 PDT, Drew Wilson, addv  
no flags  
Details | D

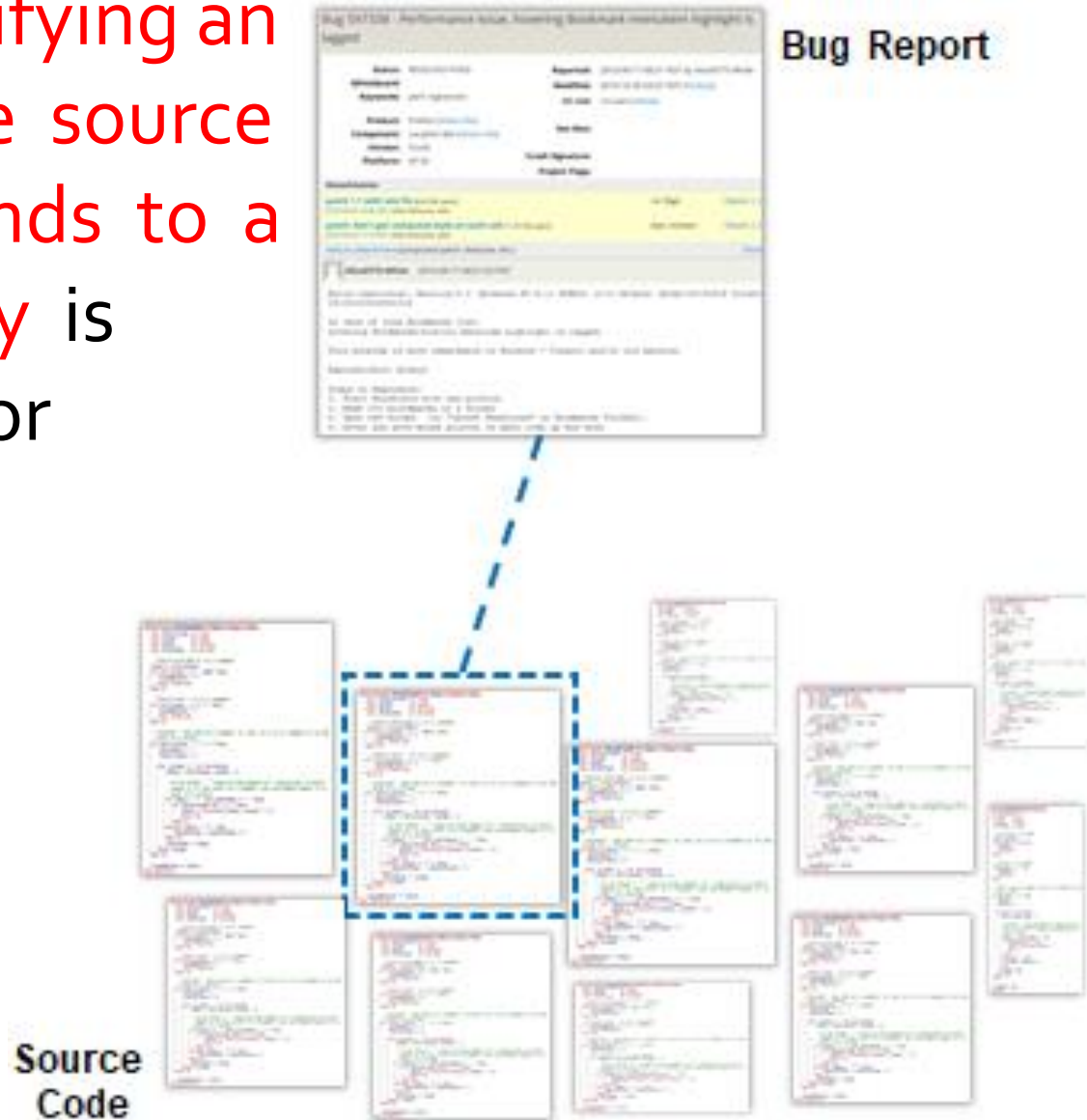
## Bug Report





# Concept (Feature) Location “Definition”\*

- The activity of identifying an initial location in the source code that corresponds to a specific functionality is known as concept (or feature) location



\*[Biggerstaff'94, Rajlich'02]

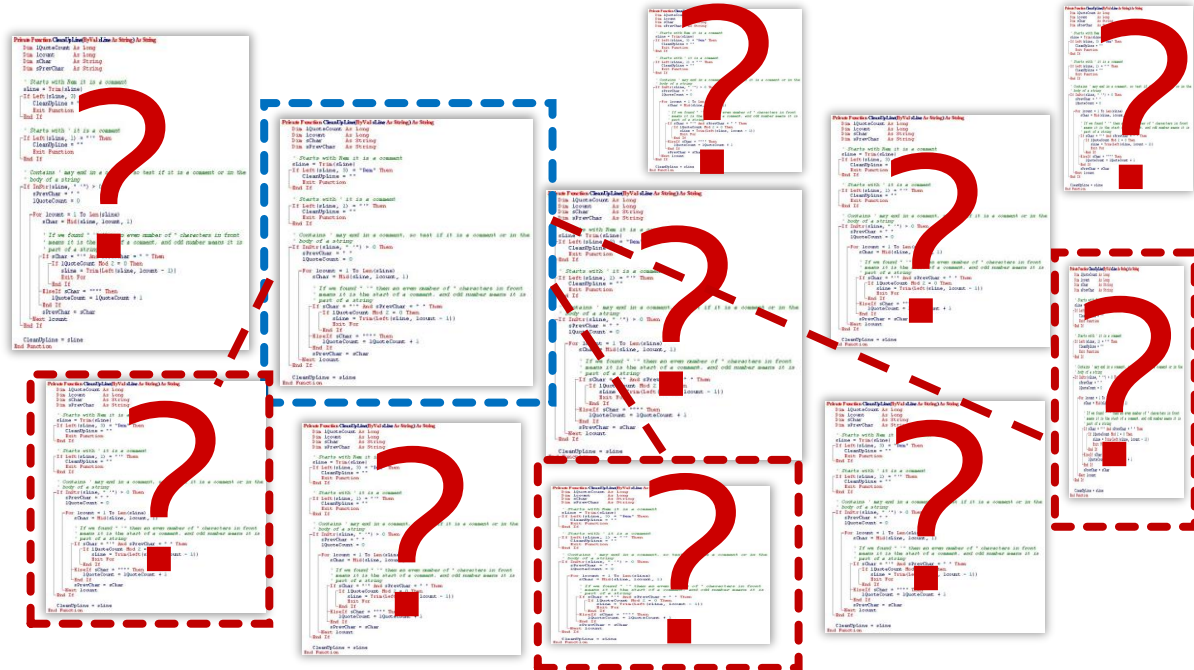
# Concept (Feature) Location

- Concept location is needed whenever a change is to be made
- Change requests are most often formulated in terms of domain concepts
  - example: “Correct error that arises when trying to paste a text”
  - the programmer must find in the code the locations where concept “paste” is located
  - this is the start of the change

# Impact Analysis “Definition”\*

- The activity of:

- estimating what needs to be modified to accomplish a change, or
- identifying the potential consequences of a change



\*Bohner and Arnold 1996

# Strategies for Concept Location

# Strategies for Concept Location:

## Familiarize with the software

- User manual/documentation
- Domain knowledge artifacts (e.g., medical process)
- Design documents and diagrams

# Strategies for Concept Location: Software Structure Comprehension

- Developer resources/wiki
- Prerequisites/dependencies
- Development workflow
- Code organization/structure and architecture
- Testing infrastructure
- Feedback channels
- Style guidelines
- APIs
- etc.

# Strategies for Concept Location: Software Build/Run

- Build/compile/resolve dependencies
- Execute the software
  - Try different scenarios/features
- Examine log files (if any)

# Strategies for Concept Location: Examine Source Code

- Search for concepts/features
- Read existing code
  - Build a mental model of the software
- Navigate dependencies
- Browse and examine code in specific locations
- Use traceability links
- Ask existing developers familiar with the code