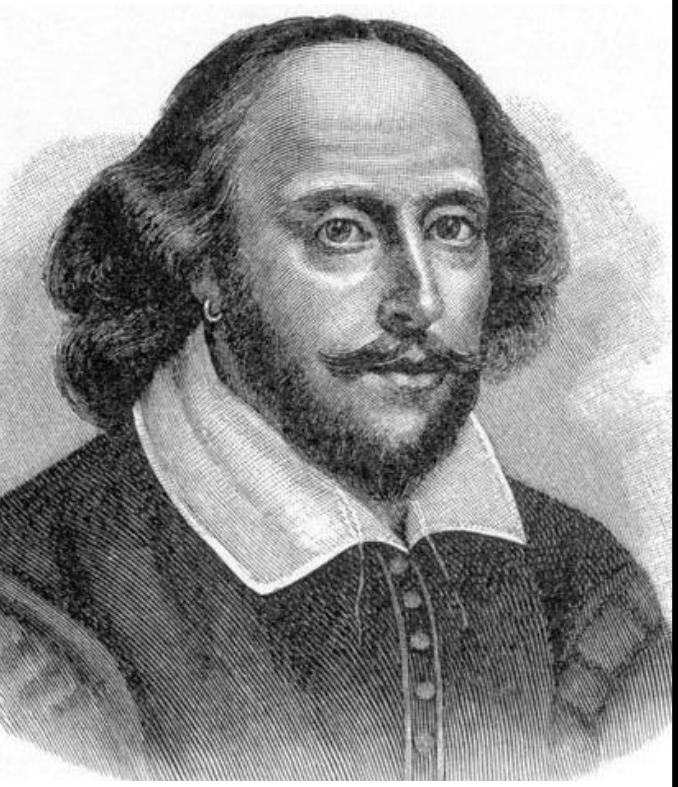
Today

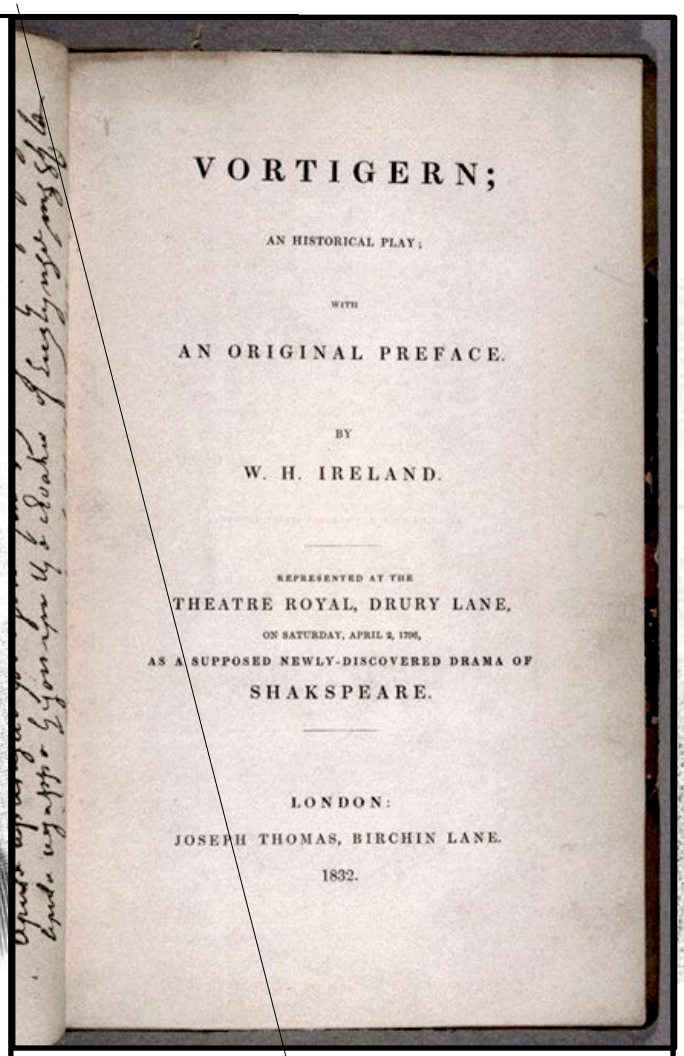
- Problem: Document Distance
  - How similar are two documents?

- Solution:
  - Algorithm idea
  - Java implementation
  - Performance measurement

Who wrote this?



William Shakespeare??



VORTIGERN;

AN HISTORICAL PLAY;

WITH

AN ORIGINAL PREFACE.

BY

W. H. IRELAND.

REPRESENTED AT THE

THEATRE ROYAL, DRURY LANE,

ON SATURDAY, APRIL 2, 1796,

AS A SUPPOSED NEWLY-DISCOVERED DRAMA OF

SHAKSPEARE.

LONDON:

JOSEPH THOMAS, BIRCHIN LANE.

1832.

***mystery* play "found" in 1796**

William Henry Ireland??

Document distance

- How similar are two documents?
  - Are two documents written by the same author?
  - Detect forgeries
  - Find plagiarism / cheating
  - Was Homer one author or many?

- What does "similar" mean?

Metrics of similarity

- Binary: (e.g., detect plagiarism)
  - Exactly same words in same order

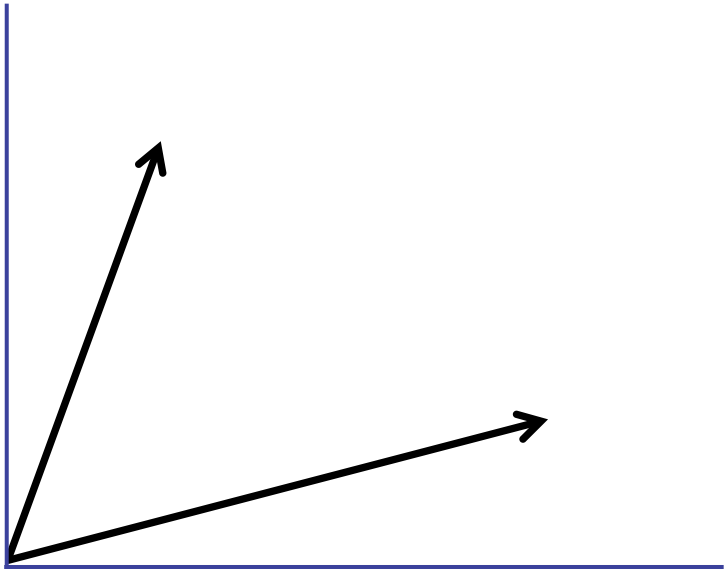
- Scalar:
  - Number of words in the same order
  - Number of shared *uncommon* words
  - Same # of words per sentence
  - Same ratio of adjectives / nouns
  - Written on similar paper / using similar ink

Vector Space Model

Strategy:

- View each document as a high-dimensional vector.

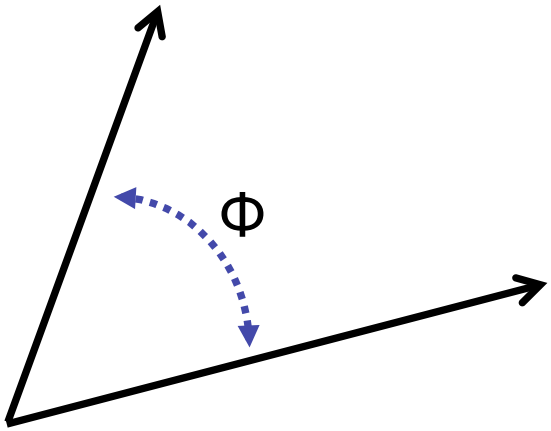
[Salton, Wang, Yang '75]



Vector Space Model

Strategy:

- View each document as a high-dimensional vector.
- The *metric of similarity* is the angle between the two vectors.



- Identical: $\Phi = 0$
- No words in common: $\Phi = \pi/2$

[Salton, Wang, Yang '75]

Vector Space Model

Document as vector:

Example 1:

"to be or not to be" = [2,1,1,2]

be	not	or	to
2	1	1	2

Vector Space Model

Example 1:

"to be or not to be" = [0,2,0,1,0,1,2]

Example 2:

"be not afraid of greatness" = [1,1,1,1,1,0,0]

afraid	be	greatness	not	of	or	to
1	1	1	1	1	0	0

Vector Space Model

Document as vector:

Example 1:

"to be or not to be" = [0,2,0,1,0,1,2]

afraid	be	greatness	not	of	or	to
0	2	0	1	0	1	2

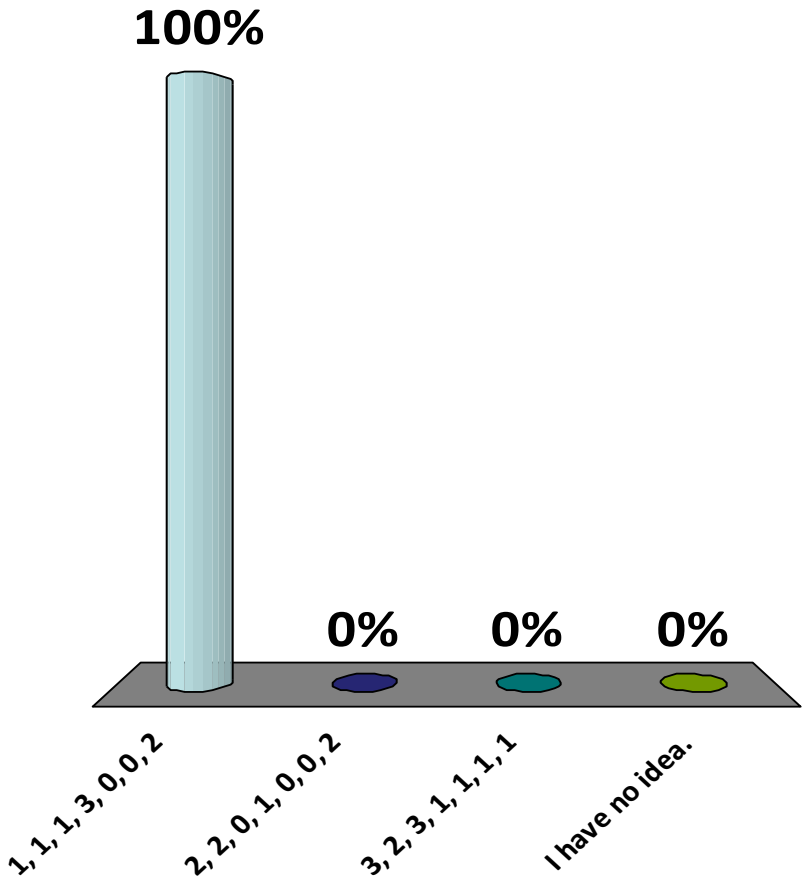
Example 3: "to be afraid, to be not afraid"

a. 1, 1, 1, 3, 0, 0, 2

✓ b. 2, 2, 0, 1, 0, 0, 2

c. 3, 2, 3, 1, 1, 1, 1

d. I have no idea.



afraid	be	greatness	not	of	or	to
?	?	?	?	?	?	?

Vector Space Model

Example 3: “to be afraid, to be not afraid”

1. [1, 1, 1, 3, 0, 0, 2]
2. [2, 2, 0, 1, 0, 0, 2]
3. [3, 2, 3, 1, 1, 1, 1]
4. I have no idea.

afraid	be	greatness	not	of	or	to
?	?	?	?	?	?	?

Vector Space Model

Dot Product:

$$v = [v_1, v_2, v_3, v_4]$$

$$w = [w_1, w_2, w_3, w_4]$$

$$v \cdot w = v_1w_1 + v_2w_2 + v_3w_3 + v_4w_4$$

Vector Space Model

Dot Product:

$$v = [v_1, v_2, \ldots, v_n]$$

$$w = [w_1, w_2, \ldots, w_n]$$

$$v \cdot w = \sum v_i w_i$$

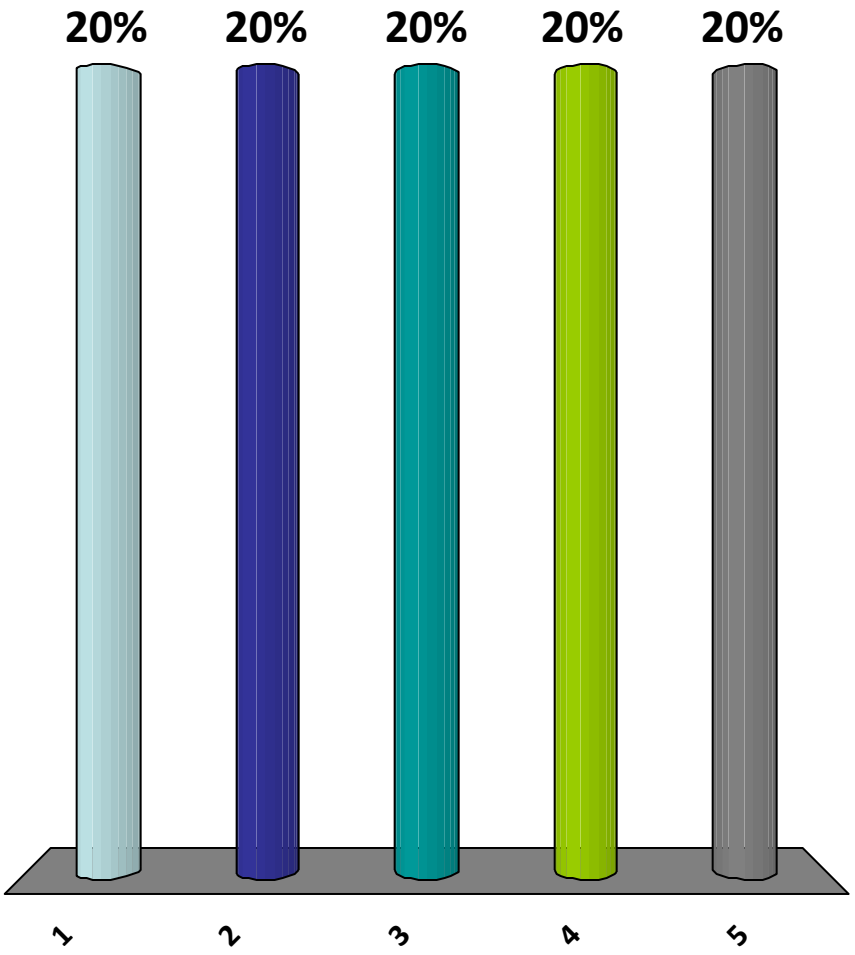
Dot product question:

v = [0, 2, 0, 1]
w = [1, 1, 1, 1]

(v·w) =

a. 1
b. 2
✓ c. 3
d. 4
e. 5

Response Counter



Vector Space Model

Norm of a vector (L2 norm):

$$|v| = \text{SQRT}(v \bullet v)$$

Example: distance between two points

$$\left|\left(x_1, y_1\right) - \left(x_2, y_2\right)\right| = \sqrt{\left(x_2 - x_1\right)^2 + \left(y_2 - y_1\right)^2}$$

Vector Space Model

Norm of a vector (L2 norm):

$$|v| = \sqrt{v \cdot v}$$

$$|v| = \sqrt{\sum_{i=1}^{n} v_i \cdot v_i}$$

Example: NORM(3, 0, 4, 0) =

SQRT(3*3 + 0*0 + 4*4 + 0*0) = 5

Vector Space Model

Law of cosines:

$$\Theta(v, w) = \cos^{-1}\left(\frac{v \cdot w}{\|v\| \cdot \|w\|}\right)$$

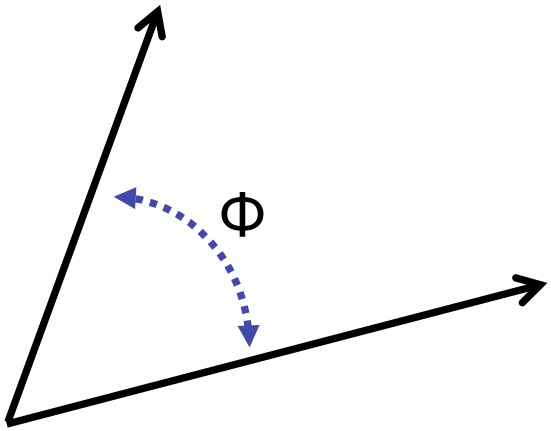
Notes:

- Φ is an angle between (0, pi)
- If (v=w), then Φ=0.
- If (v • w) = 0, then Φ=pi.

Vector Space Model

Strategy:

- View each document as a high-dimensional vector.
- The *metric of similarity* is the angle between the two vectors.



- Identical: $\Phi = 0$
- No words in common: $\Phi = \pi/2$

[Salton, Wang, Yang '75]

Compare Two Documents

Given: documents A and B

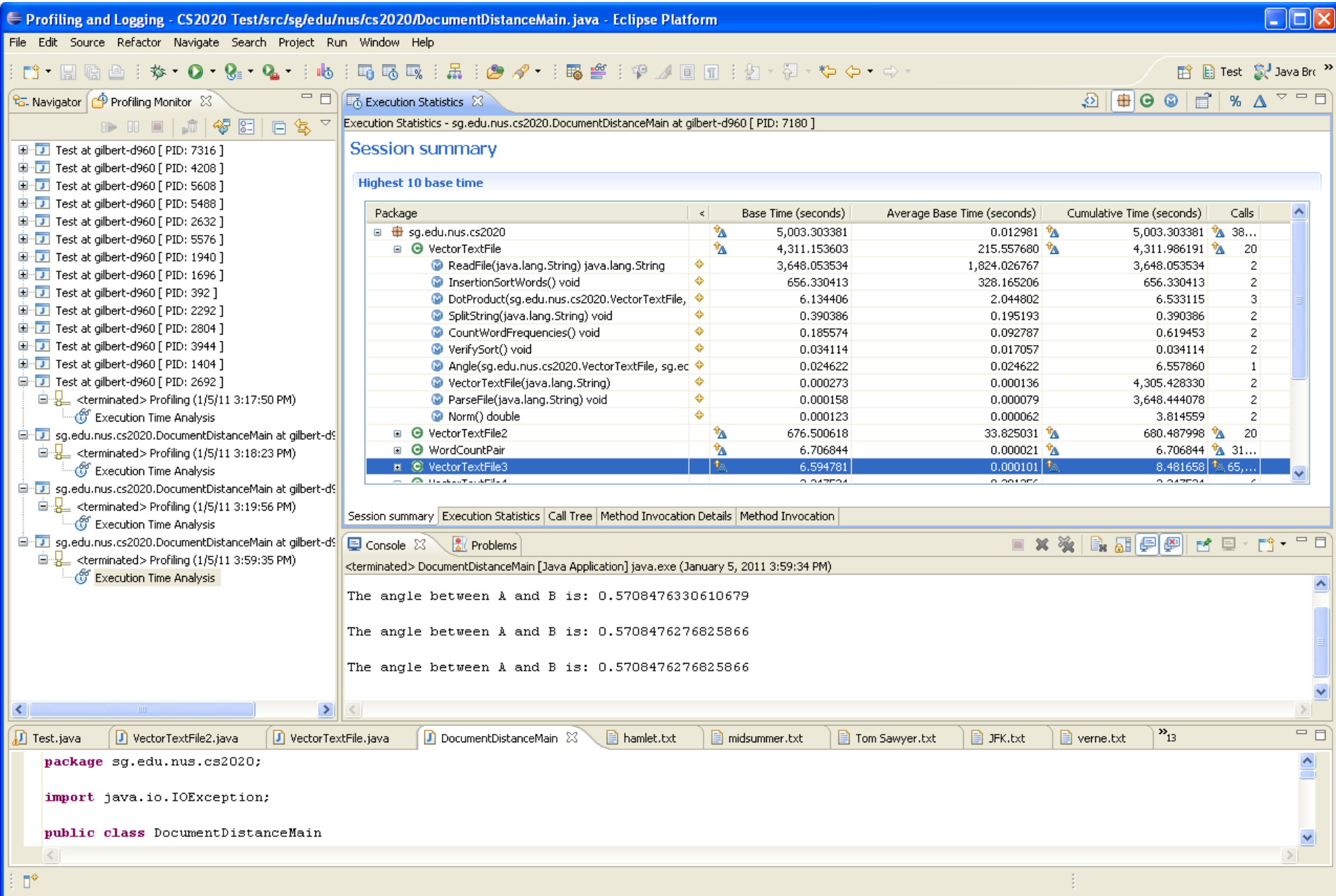
1. Create vectors v_A and v_B
2. Calculate norm: $|v_A|$
3. Calculate norm: $|v_B|$
4. Calculate dot product: $(v_A \cdot v_B)$
5. Calculate angle $\Phi(v_A, v_B)$

Performance Profiling

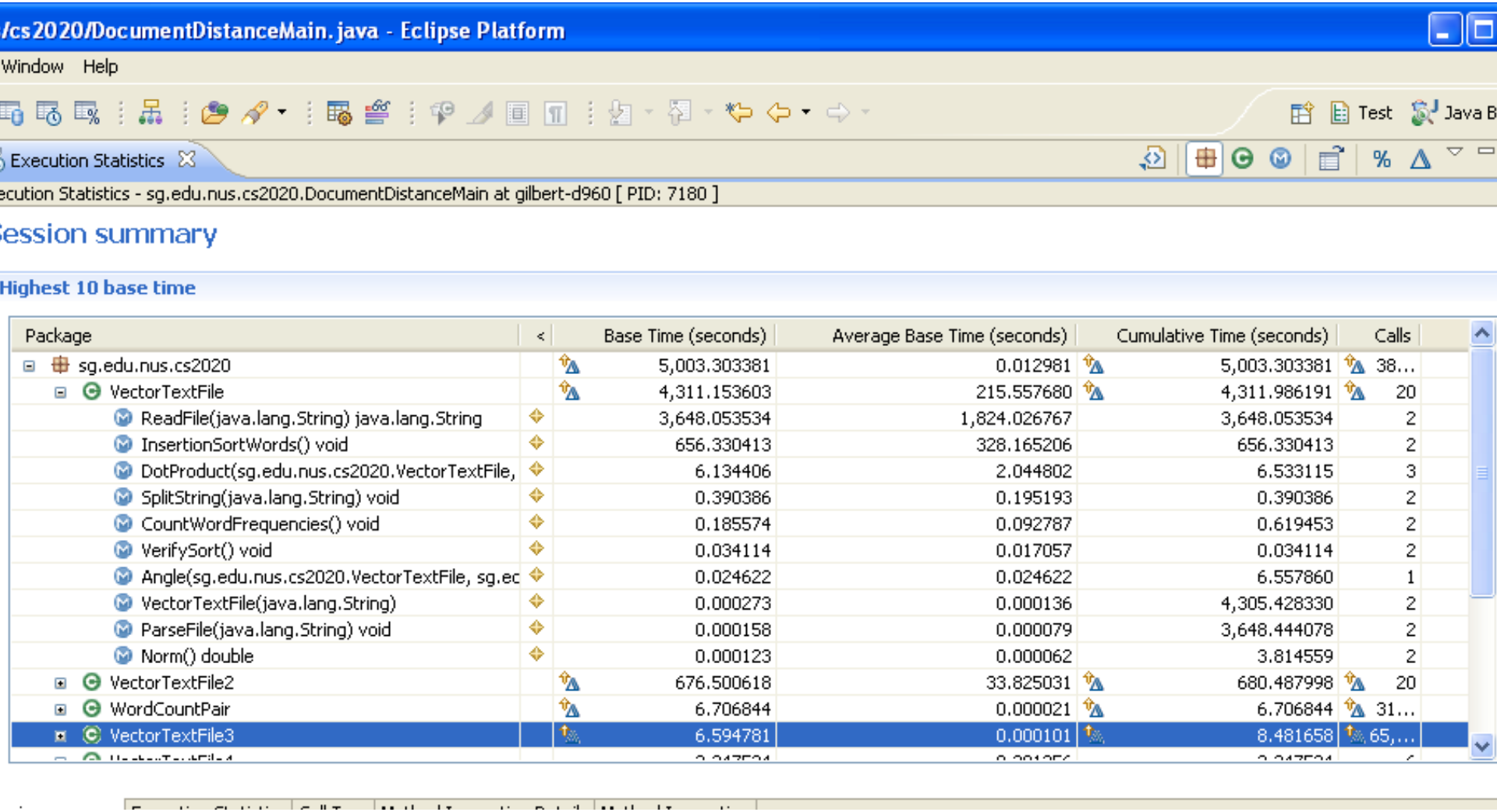
(*Dracula* vs. *Lewis & Clark*)

Step	Function	Running Time
Create vectors:	Read each file	1,824.00s
	Parse each file	0.20s
	Sort words in each file	328.00s
	Count word frequencies	0.31s
Dot product:		6.12s
Norm:		3.81s
Angle:		6.56s
Total:		**72minutes ≈ 4,311.00s**

Eclipse-TPTP



Eclipse-TPTP



Package	<	Base Time (seconds)	Average Base Time (seconds)	Cumulative Time (seconds)	Calls
sg.edu.nus.cs2020		5,003.303381	0.012981	5,003.303381	38...
VectorTextFile		4,311.153603	215.557680	4,311.986191	20
ReadFile(java.lang.String) java.lang.String		3,648.053534	1,824.026767	3,648.053534	2
InsertionSortWords() void		656.330413	328.165206	656.330413	2
DotProduct(sg.edu.nus.cs2020.VectorTextFile,		6.134406	2.044802	6.533115	3
SplitString(java.lang.String) void		0.390386	0.195193	0.390386	2
CountWordFrequencies() void		0.185574	0.092787	0.619453	2
VerifySort() void		0.034114	0.017057	0.034114	2
Angle(sg.edu.nus.cs2020.VectorTextFile, sg.ec		0.024622	0.024622	6.557860	1
VectorTextFile(java.lang.String)		0.000273	0.000136	4,305.428330	2
ParseFile(java.lang.String) void		0.000158	0.000079	3,648.444078	2
Norm() double		0.000123	0.000062	3.814559	2
VectorTextFile2		676.500618	33.825031	680.487998	20
WordCountPair		6.706844	0.000021	6.706844	31...
VectorTextFile3		6.594781	0.000101	8.481658	65,...

Performance Profiling

(*Dracula* vs. *Lewis & Clark*)

Step	Function	Running Time
Create vectors:	Read each file	1,824.00s
	Parse each file	0.20s
	Sort words in each file	328.00s
	Count word frequencies	0.31s
Dot product:		6.12s
Norm:		3.81s
Angle:		6.56s
Total:		**72minutes ≈ 4,311.00s**

Performance Profiling

(*Dracula* vs. *Lewis & Clark*)

Version	Change	Running Time
Version 1		4,311.00s
Version 2	Better file handling	676.50s
Version 3	Faster sorting	6.59s
Version 4	No sorting!	2.35s

- Version 4 will be released later in the semester…

Performance Profiling

(*Dracula* vs. *Lewis & Clark*)

Step	Function	Running Time
Create vectors:	Read each file	1,824.00s
	Parse each file	0.20s
	Sort words in each file	328.00s
	Count word frequencies	0.31s
Dot product:		6.12s
Norm:		3.81s
Angle:		6.56s
Total:		**72minutes ≈ 4,311.00s**

ReadFile (excerpt)

```
// Open the file as a stream and find its size
inputStream = new FileInputStream(fileName);
iSize = inputStream.available();

// Read in the file, one character at a time, normalizing as we go.
for (int i=0; i<iSize; i++)
{
    // Read a character
    char c = (char)inputStream.read();

    // Ensure that the character is lower-case
    c = Character.toLowerCase(c);

    // Check if the character is a letter
    if (Character.isLetter(c))
    {
        strTextFile = strTextFile + c;
    }
    // Check if the character is a space or an end-of-line marker
    else if (((c == ' ') || (c == '\n')) && (!strTextFile.endsWith(" ")))
    {
                strTextFile = strTextFile + ' ';
    }
}
```

String Problem!

What happens when:

- strTextFile = strTextFile + c

1. Creates new temporary string.
2. Copies strTextFile to the new string.
3. Adds the new character *c*.
4. Reassigns strTextFile to point to the new string.

String Problem!

What happens when:

- strTextFile = strTextFile + c

1. Creates new temporary string.
2. **Copies strTextFile to the new string**.
3. Adds the new character *c*.
4. Reassigns strTextFile to point to the new string.

Copying a string of k characters takes time k!

How long does it take here to read a file containing n characters?

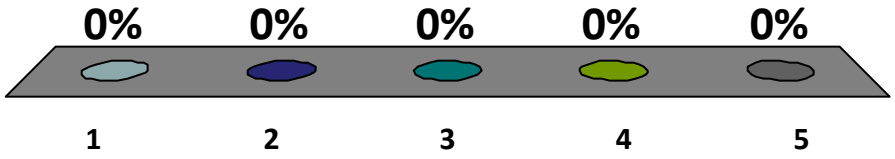
1. O(n)
2. O(n log n)
3. ✓ $O(n^2)$
4. $O(2^n)$
5. Big-O notation?

```
// Open the file as a stream and find its size
inputStream = new FileInputStream(fileName);
iSize = inputStream.available();

// Read in the file, one character at a time, normalizing as
for (int i=0; i<iSize; i++)
{
    // Read a character
    char c = (char)inputStream.read();

    // Ensure that the character is lower-case
    c = Character.toLowerCase(c);

    // Check if the character is a letter
    if (Character.isLetter(c))
    {
        strTextFile = strTextFile + c;
    }
    // Check if the character is a space or an end-of-line ma
    else if (((c == ' ') || (c == '\n')) && (!strTextFile.end
    {
            strTextFile = strTextFile + ' ';
    }
}
```



Response Counter

String Problem!

How long to read in a file of n characters?.

$$1 + 2 + 3 + 4 + \ldots + n = n(n+1)/2 = \Theta(n^2)$$

Very, very, very slow!

Fix the string problem!

```
// Open the file as a stream and find its size
inputStream = new FileInputStream(fileName);
iSize = inputStream.available();

// Initialize the char buffer to be arrays of the appropriate size.
charBuffer = new char[iSize];

// Read in the file, one character at a time, normalizing as we go.
for (int i=0; i<iSize; i++)
{
    // Read a character
    char c = (char)inputStream.read();

    // Ensure that the character is lower-case
    c = Character.toLowerCase(c);

    // Check if the character is a letter
    if (Character.isLetter(c))
    {
        charBuffer[iCharCount] = c;
        iCharCount++;
    }
    // Check if the character is a space or an end-of-line marker
    else if (((c == ' ') || (c == '\n')) && (!strTextFile.endsWith(" ")))
    {
        charBuffer[iCharCount] = ' ';
        iCharCount++;
    }
}
```

Performance Profiling, V2

(*Dracula* vs. *Lewis & Clark*)

Step	Function	Running Time
Create vectors:	Read each file	1.09s
	Parse each file	3.68s
	Sort words in each file	332.13s
	Count word frequencies	0.30s
Dot product:		6.06s
Norm:		3.80s
Angle:		6.06s
Total:		**11minutes ≈ 680.49s**

Goals for the Semester

Algorithms:

- Design of efficient algorithms
- Analysis of algorithms

Implementation:

- Solve real problems
- Analyze and profile performance
- Improve performance via better algorithms

Document Distance

(*Dracula* vs. *Lewis & Clark*)

Version	Change	Running Time
Version 1		4,311.00s
Version 2	Better file handling	676.50s
Version 3	Faster sorting	6.59s
Version 4	No sorting!	2.35s

For next time…

Friday lecture:

- Java introduction
- Object-oriented programming

Friday problem session:

- Example 2: Elevators!

Discussion Groups:

- None this week. Sign up in CORS.

Problem Set 1:

- Released. Due next week.

Administrative Details

Registration:

1. If you are not currently registered (via CORS), talk to me.
2. Register for "tutorial" session on CORS.
3. Register for "recitation" on CORS.
4. Join Facebook group

Check your e-mail:

- Invitation to Coursemology
- Invitation to NB