

UNIVERSIDAD DE GRANADA
E.T.S.I INFORMÁTICA Y TELECOMUNICACIÓN



Recuperación de Información (RI)

Preprocesado de documentos

Parte I. Parser de documentos con TIKKA

Curso 2017-2018

Cuarto Curso del Grado en Ingeniería Informática

María Camarero Granados


Javier Gómez Luzón

Francisco Porcel Molina

- Tablas con el nombre del fichero, tipo, codificación, idioma y los enlaces de cada uno de los documentos:

Título	<u>Kaksi husaaria</u>
Nombre del fichero	fin.es.html
Tipo	application/xhtml+xml
Codificación	UTF-8
Idioma	fi
Enlaces	<link href="http://purl.org/dc/terms/" rel="schema.DCTERMS"></link> <link href="http://id.loc.gov/vocabulary/relators/" rel="schema.MARCREL"></link>

Título	<u>Quijote</u>
Nombre del fichero	quijote.epub
Tipo	application/epub+zip
Codificación	windows-1252
Idioma	en
Enlaces	Full Size Full Size Full Size CHAPTER LXXIII CHAPTER LXXIV Full Size Full Size Full Size Full Size

	
--	--

Título	<u>Le vieux muet</u>
Nombre del fichero	The Project Gutenberg EBook of Le vieux muet, by Jean-Baptiste Caouette.txt
Tipo	text/plain
Codificación	UTF-8
Idioma	fr
Enlaces	

- Como generar un fichero que contenga la ocurrencia de cada uno de los términos ordenados decrecientemente en función de la frecuencia.

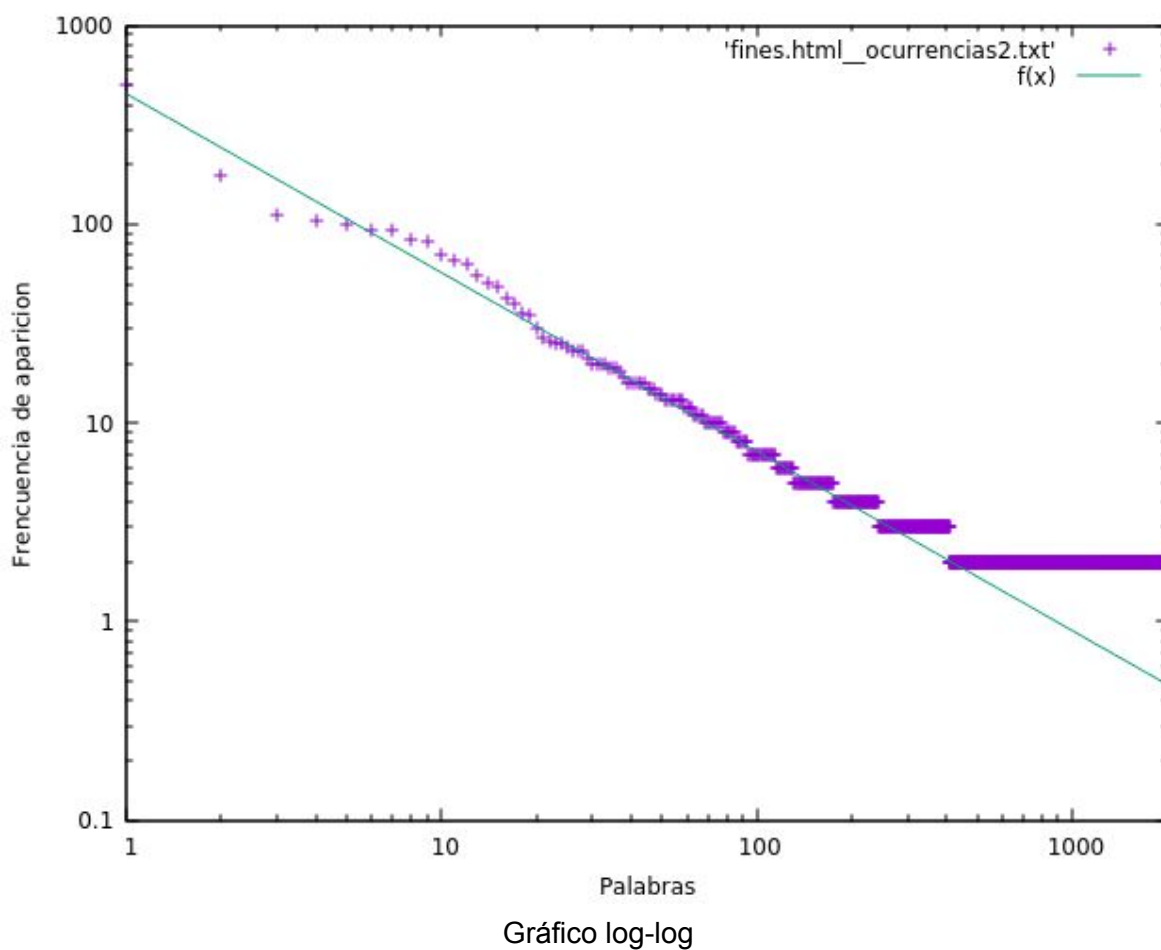
Primero se le pasa un archivo tipo File a la función process. Dentro parseamos a String con la función tika.parseToString(). Recorremos el String letra a letra, mientras no encontremos un espacio, coma, exclamación...iremos añadiendo un a un string vacío ese carácter. Cuando encontremos alguno de esos caracteres añadiremos al hashMap la nueva palabra (en caso de que no la hayamos registrado ya), en caso de que esté ya registrada incrementaremos el contador que llevamos de la palabra.

Después crearemos el fichero txt para escribir las ocurrencias de cada palabra. Utilizaremos la función nextWord, a la que le pasaremos el hashMap, que buscará la palabra con el número de ocurrencias más alto y devolverá esa palabra. Escribiremos esa palabra transformándola a minúsculas y el número de ocurrencias en el fichero y la borraremos. Esto lo haremos en bucle hasta que la función nextWord devuelve NULL, que significa que el hashMap está vacío. Para generar el fichero

para las gráficas, sustituimos la palabra por un contador que se irá incrementando para indicar el número de palabra.

- **Gráficas** donde se presentan en el eje de las X los términos ordenados en orden decreciente de frecuencia y en el eje de las Y la frecuencia de los mismos.

Kaksi husaaria:



$k = 453.678$
 $m = 0.900513$

Quijote:

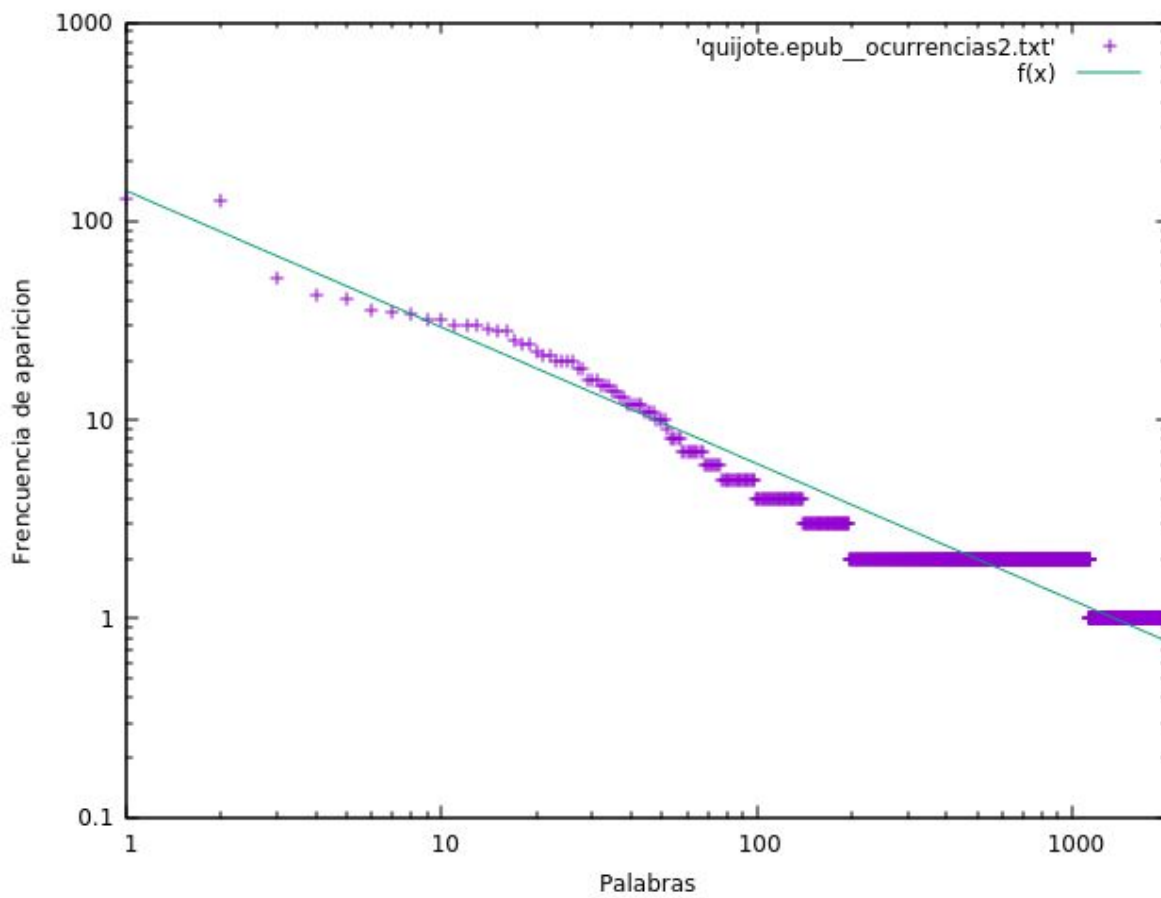
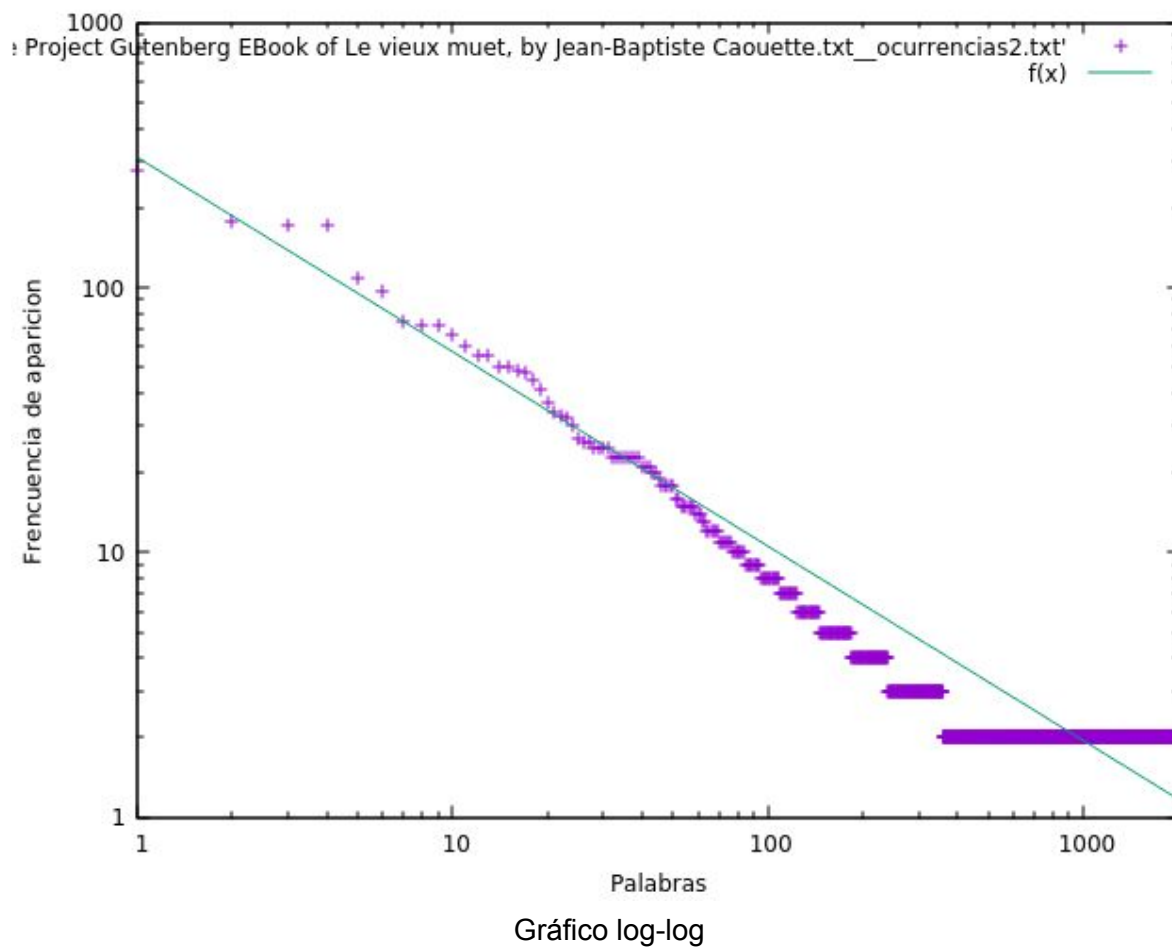


Gráfico log-log

$k = 142.326$
 $m = 0.686726$

Le vieux muet:



$k = 309.815$
 $m = 0.733628$

• Trabajo en grupo:

María Camarero Granados: Realización de la documentación

Javier Gómez Luzón: Implementación de Práctica1.java

Francisco Porcel Molina: Realización de Gráficas