

# Anatomia de Google



Basado en el artículo "***The Anatomy Of A Large Scale Search Engine***" escrito por los creadores de Google, Sergey Brin y Lawrence Page, y presentado en la WWW7 en 1997,

Una presentación realizada por **Jose Dueñas** para la asignatura ***Estructura de Datos II***

# índice del artículo

- Introducción
- Metas del diseño
- **Características del sistema**
- Trabajo relacionado
- **Anatomía del sistema**
- **Resultados y rendimiento.**
- Conclusiones
- Trabajo futuro
- Referencias

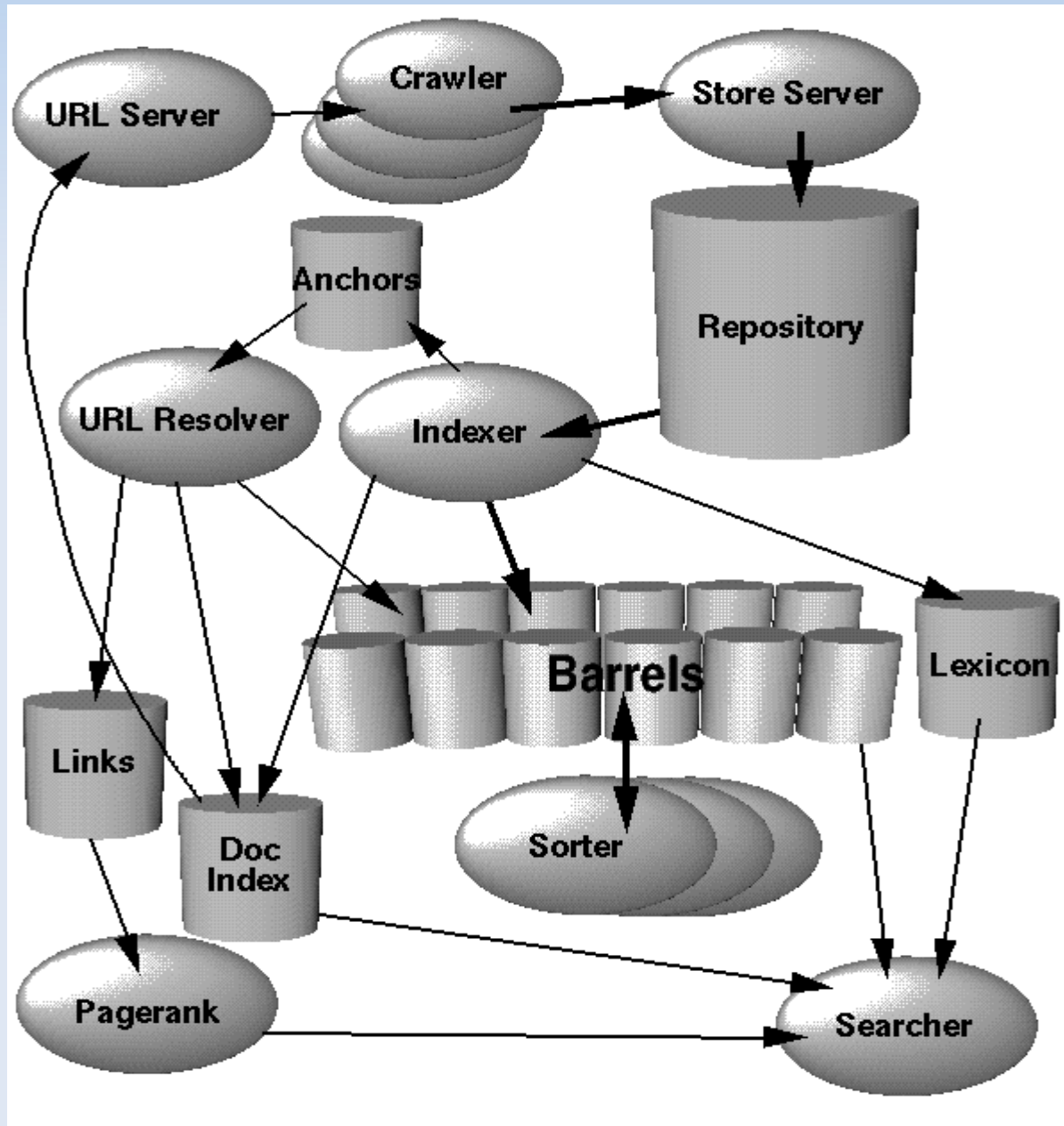
INDICE DEL ARTÍCULO

# ¿Qué es Google?

- Es un motor de búsqueda a gran escala
  - hace un uso intensivo de los enlaces
  - rastrea e indexa eficientemente la Web
  - sus resultados son mejores que otros motores
  - googol =  $10^{100}$

Actualmente es mas que un simple motor de búsqueda, aglutinando muchos otros servicios. Su objetivo según sus creadores es indexar toda la información existente y hacerla accesible.

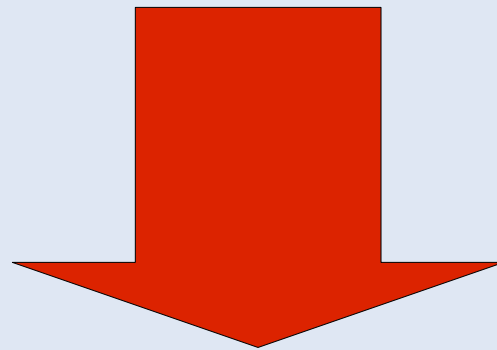
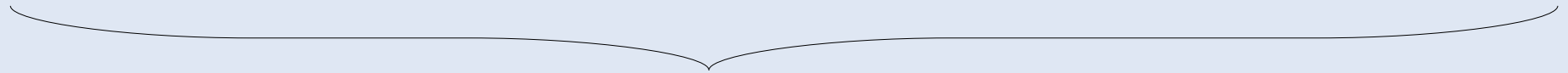
# anatomia del sistema



- Servidor URL
- Rastreador
- Servidor de almacenamiento
- Repositorio
- Textos ancla
- Servidor que resuelve URLs
- Indexador
- Cubetas
- Léxico
- Enlaces
- Indice de Documentos
- PageRank
- Clasificador
- Buscador

# los procesos

Rastreo → Indexado → Clasificación



Búsqueda

# descripción general del funcionamiento

## 1. Proceso de Rastreo (URL server – Rastreador - Storeserver)

1. URLserver envía listras de URLs a los Rastreadores.
2. Las webs son almacenadas en el storeserver
3. El storeserver las comprime y guarda en el Repositorio

## 2. Proceso de Indexación (Indexador - Clasificador)

1. El indexador las descomprime, las interpreta y asigna un docID. La web es convertida a un cjto. de ocurrencias (hits)
2. El indexador distribuye los hits en cubetas, creando un indice.
3. El indexador guarda info sobre los links en el archivo Ancla.
4. El URLresolver lee el Ancla y convierte URL relativos a absolutos que son pasados a docIDs
5. Se genera una bd con pares de docIDs (lo usará PageRank)

# descripción general del funcionamiento II

## 3. Proceso de Clasificación (Sorter)

1. El sorter reordena las cubetas (ordenadas por docID) y las ordena por wordID generando un índice invertido.
2. El sorter produce una lista de wordID y desplazamientos al índice invertido.
3. El DumpLexicon toma ésta lista y el léxico producido por el indexador y genera un nuevo léxico que será usado por el buscador.

## 4. Proceso de Búsqueda (searcher)

1. El buscador es invocado por el servidor web y usa el léxico construido por DumpLexicon junto al índice invertido y los PageRanks para resolver las búsquedas.

# las estructuras de datos

- BigFiles
- Repositorio
- Document Index
- Léxico
- Hit lists
- Índice
- Índice Invertido

# las estructuras de datos (II)

- **Big Files**

- un BigFile es un archivo que puede ocupar varios sistemas de archivos
- se puede direccionar con 64 bits
- distribución del archivo en varios sistemas gestionada por su propio Biblioteca
- ésta biblioteca permite al programador manejar los BigFiles como archivos normales
- soporta opciones básicas de compresión

# las estructuras de datos (III)

## ▪ Repositorio

- contiene el HTML completo de cada pag.
- compromiso entre velocidad y ratio de compresión
- se escoge zlib (3 a 1) sobre bzip (4 a 1)
- no requiere de otras EEDD para acceder a él
- se puede reconstruir todo a partir de él

Repository: 53.5 GB = 147.8 GB uncompressed

sync	length	compressed packet
sync	length	compressed packet

...

Packet (stored compressed in repository)

docid	ecode	urlen	pagelen	url	page
-------	-------	-------	---------	-----	------

# las estructuras de datos (IV)

## ■ Document Index

- guarda información sobre cada página
- es un archivo secuencial indexado ISAM ordenado por docID
- contiene varias estadísticas
  - puntero al repositorio, y si ha sido rastreado, puntero al archivo de tam. variable docinfo que contiene URL y título del documento.
- estructura de datos compacta
- podemos acceder a un registro en tan solo un acceso a disco.

# estructuras de datos (V)

- **Léxico**

- puede manejarse en memoria a un precio razonable
  - actualmente 256 MB
  - contiene 14 millones de palabras
  - 2 partes
    - una lista de palabras concatenadas y separadas entre ellas por NULL
    - una tabla de dispersión de punteros

# las estructuras de datos (VI)

- **Hit Lists**

- incluye información sobre la posición de la letra y si está o no en mayúscula
- ocupa la mayor parte del espacio usado por los índices
- 3 alternativas de codificación: simple, Huffman, optimizada manualmente
- codificación manual usa 2 bytes para cada hit

# las estructuras de datos (VII)

- **Hit Lists (2)**

Hit: 2 bytes

plain:	cap:1	imp:3	position: 12	
fancy:	cap:1	imp = 7	type: 4	position: 8
anchor:	cap:1	imp = 7	type: 4	hash:4   pos: 4

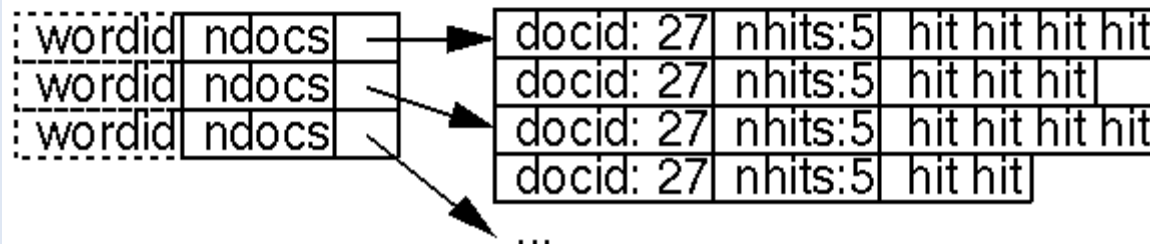
Forward Barrels: total 43 GB

docid	wordid: 24	nhits: 8	hit hit hit hit
	wordid: 24	nhits: 8	hit hit hit hit
	null wordid		
docid	wordid: 24	nhits: 8	hit hit hit hit
	wordid: 24	nhits: 8	hit hit hit hit
	wordid: 24	nhits: 8	hit hit hit hit
	null wordid		

...

Lexicon: 293MB

Inverted Barrels: 41 GB



# las estructuras de datos (VIII)

## ■ Índice

- parcialmente ordenado
- distribuido en 64 cubetas
- cada cubeta contiene un rango de wordID
- requiere mas espacio pero ahorra tiempo en la clasificación
- cada wordID se almacena como una diferencia relativa al mínimo wordID de la cubeta

# las estructuras de datos (IX)

- **Índice invertido**

- usa también 64 cubetas pero ya procesadas por el clasificador
- para cada wordID, el Léxico contiene un puntero a la cubeta correspondiente a la palabra
- el puntero apunta a una lista de docIDs junto con sus hitlists

# el sistema de ranking

- Se tiene en cuenta:
  - posición, tamaño, mayúsculas o no
  - proximidad
  - PageRank
- En general se combinan éstas técnicas de forma equitativa.
- El PageRank es una función de distribución de la probabilidad (la suma de los PageRanks de todas las pág. suman 1). Tiene muy en cuenta la estructura de enlaces y representa lo importancia de la pág..

# requerimientos de almacenaje

- los datos del repositorio son comprimidos
- se usa alrededor de 55GB para el motor de búsqueda
- la mayoría de las búsquedas pueden resolverse solo con el índice corto invertido
- con una compresión mayor, un motor de búsqueda de calidad ocuparía sólo 7GB

# rendimiento del sistema

- lleva 9 días decargar 26 millones de páginas
- 48.5 páginas por segundo
- El indexador y el rastreador pueden ejecutarse simultáneamente
- El indexador procesa a una velocidad de 54 páginas por segundo
- Los clasificadores procesan el total de las páginas con 4 PC en paralelos en tan sólo 24 horas

# conclusiones

- Motor de búsqueda escalable
- Resultados de alta calidad
- Tecnicas importantes usadas:
  - PageRank
  - Texto Ancla
  - Información de proximidad
- Es toda una completa arquitectura

# recursos y material usado

- ***The Anatomy of a Large-Scale Hypertextual Web Search Engine***, Sergey Brin and Lawrence Page -  
<http://infolab.stanford.edu/~backrub/google.html>
- ***Cómo funciona Google***, Richie Adler  
<http://www.malditainternet.com/node/60>

Pódeis obtener la traducción del artículo ***The Anatomy of a Large-Scale Hypertextual Web Search Engine*** en mi web:

<http://www.joseduenas.com>



Licencia Creative Commons