Elias-γ (Gamma) Code

To encode K:

$Kd = \lfloor \log_2 K \rfloor$, Kd es el mas grande que cumple $2^{Kd} \leq K$

$Kr = K - 2^{\lfloor \log_2 K \rfloor}$

Codificación:

- Kd en unario
- Kr en binario usando Kd digitos

Bits que requiere:

$\lfloor \log_2 K \rfloor + 1$ bits para Kd en unario
$\lfloor \log_2 K \rfloor$ bits para Kr en binario
→ Total $2\lfloor \log_2 K \rfloor + 1$ bits

Elias δ (Delta) Code

Todo nro K:

$Kd = \lfloor \log_2 K \rfloor$

$Kr = K - 2^{\lfloor \log_2 K \rfloor}$

$Kd+1$ se codifica en Elias Gamma

Kr se codifica en binario usando Kd dígitos

$Kdd = \lfloor \log_2 (Kd+1) \rfloor$ ⇝ en unario

$Kdr = (Kd+1) - 2^{\lfloor \log_2 (Kd+1) \rfloor}$ → en binario usando Kdd dígitos

Error en la slide del 45rto!

Ej. $K = 1023$

$Kd = \lfloor \log_2 K \rfloor = 9$ $\quad (2^9 = 512,\ 2^{10} = 1024)$

$Kr = K - 2^{\lfloor \log_2 K \rfloor} = 1023 - 512 = 511$

$Kd+1 = 10$ se codifica en Gamma

$Kr = 511$ se codifica en binario usando $Kd = 9$ dígitos

$Kdd = \lfloor \log_2 (Kd+1) \rfloor = 3$ $\quad 2^3 = 8 \leq 9$

$Kdr = (Kd+1) - 2^{\lfloor \log_2 Kd+1 \rfloor} = 10 - 2^3 = 2$

1110 010 111111111

1110: $Kdd = 3$ en unario

010: $Kdr = 2$ en binario usando $Kdd = 3$ dígitos

111111111: $Kr = 511$ en binario usando $Kd = 9$ dígitos

Shannon - Teoría Información

Cantidad de información de un símbolo s_i que se emite con probabilidad p_i $\quad I(s_i) = -\log_2 p_i$

$p_i = 2^{-I(s_i)}$

- Si $p_i = 1 \Rightarrow I(s_i) = 0$
- Si $p_i \to 0 \Rightarrow I(s_i) \to \infty$

Entropía o cantidad de información de una distribución de probabilidad

$$H(P) = -\sum_{i=1}^{n} p_i \log_2 p_i = \sum_{i=1}^{n} p_i I(s_i)$$

↓ promedio de la cantidad de información de los eventos ponderados con sus probabilidades

Ej: $P = \{\text{cara, cruz}\} = \{\frac{1}{2}, \frac{1}{2}\}$

$$H(P) = -\frac{1}{2}\log_2\frac{1}{2} - \frac{1}{2}\log_2\frac{1}{2} = -\frac{1}{2} - \frac{1}{2} = \boxed{1 \text{ bit}}$$

↳ con 2 eventos equiprobables, la cantidad de información de saber el resultado por adelantado es 1 bit.

Ej: moneda trucada 99% cara

$$P = \{\frac{99}{100}, \frac{1}{100}\}$$

$H(P) = 0'08$ bits

La cantidad de información de saber la salida por adelantado es mucho menor.

La longitud ideal del código de un símbolo l_x y su probabilidad $Pr(x)$ idealmente deben seguir la relación de Shannon

$$\boxed{l_x = -\log_2 Pr(x)} \quad \boxed{Pr(x) = 2^{-l_x}}$$

$Pr(x)=1 \Rightarrow$ Sólo un símbolo posible $l_x = 0$

$Pr(x) \to 0 \Rightarrow l_x \to \infty$

- N símbolos con probabilidad uniforme, i.e., $Pr(x) = \frac{1}{N}$ $l_x = -\log_2 \frac{1}{N}$, es decir, longitud cte para todos los símbolos. → Codificación binaria con extensión fija y el mismo para todos los valores posibles a codificar
- Codificación unaria

  $l_x = x$ (la longitud del código es igual al valor que se codifica).

$$Pr(x) = 2^{-x}$$

Codificación ideal para una distribución de probabilidad

$$\{ \tfrac{1}{2}, \tfrac{1}{4}, \tfrac{1}{8}, \tfrac{1}{16} \dots \}$$

El código 10 ocupa el 50% de los recs, el código 110, 25% recs, 1110, 12'5% de los recs, etc.

Codificación gamma γ

$$Pr(x) = 2^{-l_x} = 2^{-(1+2\log_2 x)} = \frac{1}{2x^2}$$

$$\frac{1}{2^{(1+2\log_2 x)}} = \frac{1}{2\cdot 2^{2\log_2 x}} = \frac{1}{2\cdot 2^{\log_2 x^2}} = \frac{1}{2x^2}$$

Distribución probabilidad ideal

$\{\frac{1}{2}, \frac{1}{8}, \frac{1}{16}, \ldots\}$

El valor $x=1$ (código 10) $\frac{1}{2}$ veces, valor $x=2$ (código 100) $\frac{1}{8}$ veces, valor $x=3$ (código 101) $\frac{1}{16}$ veces

Reserva longitudes de codificación largas para probabilidades más bajas que en unario.

Codifica delta δ

$$Pr(x) = 2^{-l_x} = 2^{-(1+2\log_2 \log_2 x + \log_2 x)} =$$

$$= \frac{1}{2x(\log x)^2}$$

Para números pequeños los delta codes son más largos que los gamma codes, pero para números grandes de repente

$x = 1.000.000$ → γ code 39 bits
→ δ code 28 bits

Compression Example

(1, 2, [1, 7]) (2, 3, [6, 17, 197]) (3, 1, [1])

- [1, 7] → positions
- 1 → tf
- 2 → doc-id

D-gaps para doc-id y positions:

(1, 2, [1, 6]) (1, 3, [6, 11, 180]) (1, 1, [1])

Continuation bit + 7-byte

81 82 81 86 81 83 86 8B 01 B4 81 81 81

8 | 1 → 0000 0001

8 | B → 1000 1011

0000 0001 | 1011 0100

B = 11, 4 = 0100

1011 0100

128 + 32 + 16 + 4 = 180

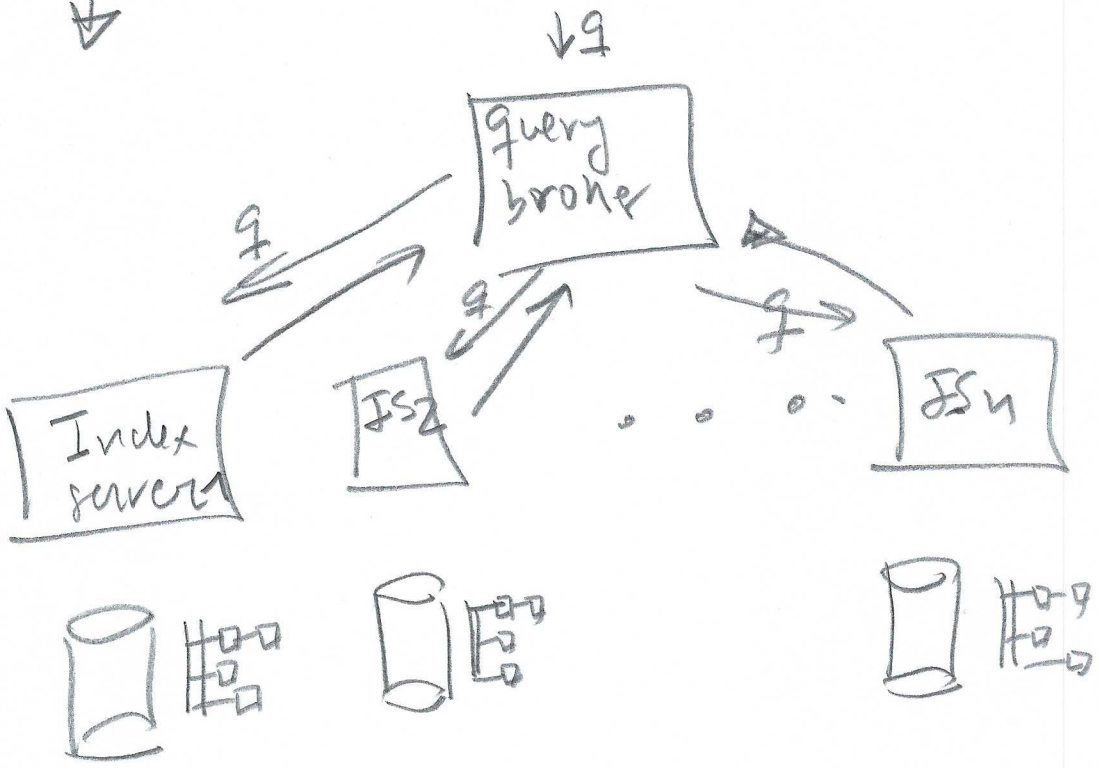
Skip Pointers

D-gaps: ~~9~~

5, 6, 6, 4, 5, 8, ...

Distributed Evaluation

- Document distribution (partitioning)
- Term " "



Term distribution: Para una query de k términos y n index servers, el número de disk seeks necesarios será $O(k)$, pero en document distribution será $O(k \cdot n)$ porque cada query term hay que procesarlo en los n servidores (index servers)

Term Distribution, el software es más complejo y se requieren más mensajes entre index servers para procesar una query, pero es más eficiente en E/S.

- Federated (Pure Distributed) Search.

Los docs y por tanto indices creados por los docs están en sitios distintos por razones de privacidad, legales, etc.

Pasamos en Document Partitioned: El broker no tiene las estadísticas globales y no tiene acceso a las colecciones. Puede hacer

- Pedir a los sitios las estadísticas de sus colecciones o hacer query sampling
- Resource (Site) selection
- Lanza la query solo a los sitios seleccionados
- Fusiona los resultados.

- Metasearch

Fusión de scores o de ranking (posiciones) de diversos search engines a los que puedes acceder por un API

Técnica de combinación.
CombMNZ, Bordeaux, etc.

Ver Aslam, Montague. Models for Metasearch, ACM SIGIR 2001

Caching de queries

Ley de Zipf (Potencia negativa)

$$f(n) = C\, n^{-\alpha} = C \cdot \frac{1}{n^{\alpha}}$$

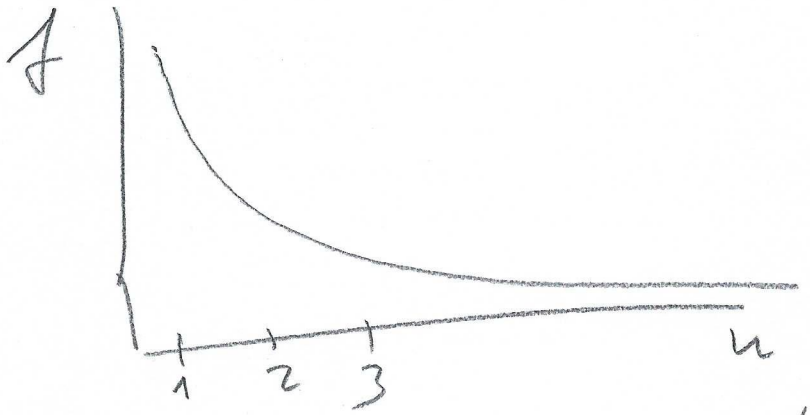
f ← frecuencia o ocurrencia query

n, ranking por frecuencia

$n=1$ query más frecuente

$n=2$ 2ª query más frecuente

etc.



Ley experimental que favorece caching