Recuperatorio Information Retrieval

Dados los siguientes documentos:

D1: Caballo Caballero Cabra

D2: Cacao Cacerola Caballo Cabalgar

D3: Cacao Cabra Cabaña

D4: Caballo Cabalgar Caballero Caballo

D5: Cabra Cacerola

Se pide:

1) Indicar el paso a paso de la construcción de un índice invertido con Front Coding parcial (n=3) indicando detalladamente la estructura resultante. Utilizar códigos gamma para la codificación de punteros. ¿Cuánto espacio ocupa el índice?

2) Resolver la consulta Caballo Caballero

3) Indicar detalladamente qué cambios propondría a la estructura generada en el punto 1 para que la consulta por frase "Caballo Caballero" no arroje falsos positivos. No es necesario que implemente los cambios en el índice, solo que detalle como quedaría con un ejemplo, pero si le resulta más fácil mostrar directamente la nueva estructura puede hacerlo también.

RESOLUCIÓN

1) Lo primero que se debe realizar para obtener el índice invertido a partir de los documentos es obtener todos los términos que aparecen en los documentos, y en cuáles aparecen. Estos sería:
    - Caballo: D1, D2, D4
    - Caballero: D1, D4
    - Cabra: D1, D3, D5
    - Cacao: D2, D3
    - Cacerola: D2, D5
    - Cabalgar: D2, D4
    - Cabaña: D3

Luego se debe ordenarlos alfabéticamente:

- Cabalgar: D2, D4
- Caballero: D1, D4
- Caballo: D1, D2, D4
- Cabaña: D3
- Cabra: D2, D3
- Cacao: D2, D3
- Cacerola: D2, D5

Ahora puedo armar el front coding parcial de los términos con n=3. Esto significa que cada 3 términos, la cantidad de caracteres que comparte con el anterior se resetean, ya que en front coding lo que se busca es ahorrar espacio reduciendo la cantidad de caracteres que se almacenan por término:

- Cabalgar (Iguales 0, Distintos 8)
- Caballero (Iguales 5, distintos 4)
- Caballo (Iguales 6, distintos 1)
- Cabaña (Iguales 0, distintos 6) Por el reset en n=3
- Cabra (Iguales 3, distintos 2)
- Cacao (Iguales 2, distintos 3)
- Cacerola (Iguales 0, distintos 8) Por el reset en n=3

Finalmente el front coding parcial con n=3 queda:

$$Cabalgar_{8}lero_{12}o_{13}Caba\tilde{n}a_{19}ra_{21}cao_{24}Cacerola_{32}$$

Ahora, para la codificación gamma lo que se realiza es el pasaje de los documentos a distancias, y luego, se los codifica en gamma y se añaden los códigos concatenados uno atrás del otro, correspondiendo a cada término en orden:

- D2, D4 = 2, 2 = 010 010
- D1, D4 = 1, 3 = 1 011
- D1, D2, D4 = 1,1,2 = 1 1 010
- D3 = 3 = 011
- D2, D3 = 2,1 = 010 1
- D2, D3 = 2, 1 = 010 1
- D2, D5 = 2, 3 = 010 011

Finalmente, la codificación queda (Los espacios son meramente visuales, en máquina iría un bit atrás del otro):

$$010\ 010_{6}1\ 011_{10}1\ 1\ 010_{15}011_{18}010\ 1_{22}010\ 1_{26}010\ 011_{32}$$

Entonces ahora el índice invertido se construye con las direcciones de los punteros a términos, y los punteros a la codificación gamma; pero como use front coding parcial también tengo que almacenar la cantidad de caracteres iguales, y los distintos (Estos últimos se pueden obviar y calcular a partir de Término Final – Cant Iguales)(La columna índice sería la posición en el índice invertido, no sería algo almacenado):

Índice	Término	Char =	Char distintos	Documentos
0	0	0	8	0
1	8	5	4	6
2	12	6	1	10
3	13	0	6	15
4	19	3	2	18
5	21	2	3	22
6	24	0	8	26

El tamaño final será de 32 bytes correspondientes a los términos más 32 bits correspondientes a los códigos gamma de los documentos; por lo que el tamaño del mismo será de 36 bytes.

Luego, si se considera el tamaño de los punteros, y de los ints almacenados en el índice, se le deben sumar 7 filas * 4 bytes * 4 datos/fila = 112 bytes

Entonces, el tamaño que representaría el índice sería de 148 bytes.

2) Para resolver la consulta caballo caballero debo recurrir a una búsqueda binaria. Lo primero que se hace es acceder al término del medio del índice, que en este caso será el índice 3 debido a que poseo 7 elementos:
   Búsqueda de caballero
   - Acceso a la pos 3 del índice (1 acceso a índice); en el índice leo el puntero al término, la cantidad de char iguales, la cantidad de char distintos, y con eso me basta ya que debo leer la cantidad de caracteres distintos desde mi puntero a términos.
   - Accedo a disco en búsqueda del término. El término obtenido es cabaña, y caballero es menor que este término. (1 acceso a disco)
   - Como caballero es menor continuo con la búsqueda binaria del lado izquierdo. Accederé al elemento en la posición 1 del índice (1 acceso a índice), leeré el puntero a término, la cantidad de char distintos, la cantida de char iguales, y como char iguales es mayor a 0, debo leer

también del término anterior su puntero a término, sus char iguales, y sus char distintos.

- Accedo a disco en búsqueda del término, leyendo primero los 5 chars del elemento 0, y luego los char distintos del elemento 1. El término obtenido es Caballero, que es el elemento buscado.
- Obtengo del índice el puntero a documentos de la posición 1, y de la posición 2 para conocer hasta donde tengo que leer. (1 acceso a índice)
- Leo las codificaciones de documentos de disco, y obtengo:
  1 011 que traducido es D1, D4. (1 acceso a disco)

  Entonces, caballero se encuentra en los documentos D1 y D4.

Búsqueda de caballo

- Se repiten los mismos pasos que para caballero, pero todos los accesos a disco ahora se encuentran en memoria. Habrá 3 accesos a índice y 2 accesos a datos en memoria.
- Como caballo es mayor que caballero me resta leer el único término que me queda del lado derecho de la posición 1. Entonces, accedo al índice en la posición 2, leo el puntero a término, los char = y los char distintos. Accedo a disco en búsqueda de los char distintos de los elementos en esta posición, y obtengo la "o". (1 acceso a índice, 1 acceso a disco).
- Concatenando con los char iguales, se obtiene el término "Caballo" que es el término buscado.
- Se accede a índice para obtener el puntero a documentos de la posición 2, y leo el puntero a posición 3 de los documentos para saber hasta dónde leer.
- Leo la codificación gamma de disco (1 acceso a disco), y obtengo:
  1 1 010, que traducido es D1, D2, D4.

  Entonces, caballo se encuentra en los documentos D1, D2 y D4.

Finalmente para terminar de resolver el query se hace un AND de ambos conjuntos, y obtenemos que el resultado final para los cuales se encuentran tanto el término caballo como el término caballero son los documentos D1, y D4.

3) Para poder permitir una consulta por frase se debe tener un índice invertido que permita búsqueda de frases por proximidad. Para esto se debe codificar los documentos de la siguiente manera:

Doc1 Freq Pos1 … Posn … Docn Freq Pos1 … Posn

Donde lo que se almacena sería el documento en el que se encuentra, la frecuencia con la cual se encuentra al término en ese documento, y las posiciones codificadas como distancias en las cuales se encuentra ese término dentro del documento.

De esa manera, para resolver la consulta de la frase “Caballo Caballero” tendría que considerar que para que el resultado sea positivo, tenga que tener a Caballo dentro de un documento en una posición i, y a Caballero también dentro del documento en una posición i+1. Si eso no se cumple, entonces el resultado es considerado negativo.

En vista de los datos del enunciado, el resultado con una consulta por frase de “Caballo Caballero” sería únicamente D1, ya que D4 tiene a caballo seguido del término cabalgar. Esto significaría que D4 sería un falso positivo si no se utiliza proximidad.