Guia 4 Hashing : Resolucion

Di Maria, Franco Martín,
Padrón : 100498
2do. Cuatrimestre de 2019
75.06 - Organización de Datos
Facultad de Ingeniería
Universidad de Buenos Aires

1. Ejercicio 1

1.1. Enunciado

Al construir una función de hashing perfecta y mínima con el método visto en clase explique por que es necesario que M sea mayor a N.

1.2. Solución

En el algoritmo de hashing perfecto y mínimo (version 2), se define: N : Cantidad de claves M : numero mayor a N, que corresponde al espacio de direcciones al cual ira el resultado de aplicar dos funciones de hashing, a cada clave .

Se enumeran las claves, y se utiliza este numero como valor de la arista, que uno los dos resultados de aplicar las dos funciones de hashing anteriores. Estos hashes son nodos del grafico.

Lo que buscamos es generar un grafo sin ciclos. Si M no fuese mayor a N, entonces siempre tendriamos al menos un ciclo. Con M mayor a N, y adecuadas funciones de hashing, podemos obtener un grafo sin ciclos. La probabilidad de ocurrencia de un ciclo, disminuye si aumentamos la cantidad de nodos (manteniendo fija la cantidad de aristas, que son las claves), es decir, aumentando M.

2. Ejercicio 2

2.1. Enunciado

Encontrar el m mínimo y los parámetros a y b de forma tal que la función de hashing ax + b mod m sea perfecta para las siguientes claves: 1,3,5,12

2.2. Solucion

3. Ejercicio 5

3.1. Enunciado

Tenemos un total de 10.000 claves de 32 bytes c/u. Si usamos el esquema FKS y la primer tabla tiene 1000 posiciones. ¿Cuánto espacio necesitamos en total para almacenar las 10.000 claves?

3.2. Solución

En el esquema FKS, el costo en espacio es 2m, siendo m la cantidad de claves.

Si m = 10000 claves, y cada clave ocupa 32 bytes, entonces el espacio de memoria que consume FKS, en este caso, es de :

$$Espacio\ total = 2{*}m{*}32\ Bytes = 2{*}10000{*}32\ Bytes = 640000\ bytes = 640\ KB$$

4. Ejercicio 6

4.1. Enunciado

Supongamos que asignamos a cada letra del alfabeto un número de la forma A=1, B=2, C=3, etc... Proponemos como función de hashing sumar el valor correspondiente a cada carácter del string y luego tomar el módulo con un cierto número primo p. Analizar la función propuesta indicando:

a) Cantidad de colisiones

b) Facilidad de encontrar sinónimos

c) Eficiencia

d) Efecto avalancha

4.2. Solución

4.2.1. A

Si P es grande, la cantidad de colisiones, sera menor.

Si P es pequeña, entonces la cantidad de colisiones sera mayor.

Si P es aproximadamente la sumatoria de todo el abecedario, aun así habrá muchas colisiones, pues si suponemos que una palabra no puede tener mas de 10 caracteres, y varios de ellos, serán probablemente vocales, entonces es probable que varias palabras colisionen, pues muchas de ellas tendrán un hash idéntico o cercano.

4.2.2. B

Como los valores de cada letra son muy cercanos entre ellos, no es muy difícil encontrar otra combinación de letras que resulte en la misma sumatoria, y por ende en el mismo hash

4.2.3. C

Como la función de hashing solo debe sumar los valores de los caracteres en el string, es eficiente en cuanto a velocidad.

4.3. D

Como las letras tienen valores muy cercanos entre ellos, y la función solo suma estos valores, y les aplica el modulo de p, entonces un cambio mínimo en la string no debería generar un gran cambio en el hash, excluyendo el caso donde p sea extremadamente grande.