Reto 5
Estructura de Datos

Jose Antonio Padial Molina

josepadial@correo.ugr.es

December 12, 2018

Contents

1 Enunciado

El reto 5 consiste en construir un esquema de hashing doble. Disponemos de una función hash.

$$h(k) = k\%M$$

Donde M es primo
y de un método de resolución de colisiones siguiendo un esquema de hashing doble a través de la función:

$$h_1(k) = [h(k) + d_1]\%M$$

$$i = 2, 3, 4, ...$$

con d_1 calculado como:

$$d_i = [a * d_{i-1} + c]\%M$$

con $d_0 = 0$

Se trata de encontrar para un valor de M una combinación de valores de a y de c que den lugar a una secuencia de d_i que no cicle antes de tiempo, es decir una secuencia de valores que garantice el acceso a todos los posibles huecos de la tabla cara a la insercion de nuevas claves. El reto tiene 2 partes:

1. Mediante un sencillo procedimiento de simulacion, hacer que el ordenador encuentre las parejas (a,c) que den lugar a un esquema de hashing valido (sucesión que no cicle antes de tiempo).
2. Extraer conclusiones del resultado, es decir dar posibles soluciones teóricas independientemente del tama no M de la tabla que tengamos (evidentemente una vez construido el procedimiento de simulacion podeis aplicarlo a diferentes valores de M y ver que saca porque eso os ayudara a extraer conclusiones generales).

2 Solución

Para ello se han implementado tres funciones y el main:

2.1 Función 1

```
int hash(int k) {
  int h;
  h=k%M;
  return h;
}
```

2.2 Función 2

```
int hashCol(int k, int i) {
  int h_i;
  h_i=(hash(k)+d(i))%M;
  return h_i;
}
```

2.3 Función 3

```
int d(int i) {
    if(i==0)
        return c%M;
    else
        return (a*d(i-1)+c)%M;
}
```

2.4 Main

```
int main() {
    int vector[M];
    for(a=1;a<30;a++){
        for(c=1;c<10;c++){
            int contador=0;
            for(int i=0;i<M;i++)
                vector[i]=-1;
            for ( int i=2; contador< M && vector[d(i)]!=1;i++){
                vector [d(i)]=1;
                contador++;
            }
            if(contador == M)
            cout<<"Si: "<<a<<","<<c<<endl;
        }
    }
    return 0;
}
```

Debemos encontrar para qué pareja (a,c) el método de resolución de colisiones d_i asigna posiciones distintas. Tras ejecutar el programa saco como conclusión que si a y c no hacen que el valor d_i sea constante se garantiza el acceso a todos los posibles huecos de la tabla. No cicla antes de tiempo.

2.5 Ejemplos

Ejemplos de valores: (1,1),(2,2) entre otros. Ejemplos que hacen d_i constante: (1,0).