INSTITUTO POLITÉCNICO NACIONAL
UNIDAD INTERDISCIPLINARIA DE INGENIERÍA CAMPUS ZACATECAS

Profesor Roberto Oswaldo Cruz Leija

Análisis de Algoritmos

Unidad 2

Algoritmo Karp-Rabin

Leslie Arellano Covarrubias

3CM1

Fecha de Entrega

14/11/2019

Introducción

El programa que se presenta a continuación es una implementación del algoritmo Karp-Rabin, escrito en lenguaje C y con la utilización de cadenas de caracteres y ciclos para la búsqueda de un patrón propuesto.

Objetivo

Entender el funcionamiento del algoritmo Karp-Rabin para realizar su implementación y encontrar distintas soluciones al problema.

Marco Teórico

El algoritmo Karp-Rabin es un Algoritmo de búsqueda de subcadenas simple enunciado por Michael Oser Rabin y Richard Manning Karp en 1987. Se basa en tratar cada uno de los grupos de m caracteres del texto (siendo m el número de símbolos del patrón) del texto como un índice de una tabla de valores hash o de dispersión, de manera que si la función hash de los m caracteres del texto coincide con la del patrón es posible que hayamos encontrado un acierto. Para verificar hay que comparar el texto con el patrón.

La función hash tiene la forma d(k)=xk % d donde d es un número primo grande que será el tamaño de la tabla de dispersión y xk se calcula de la forma indicada más abajo.

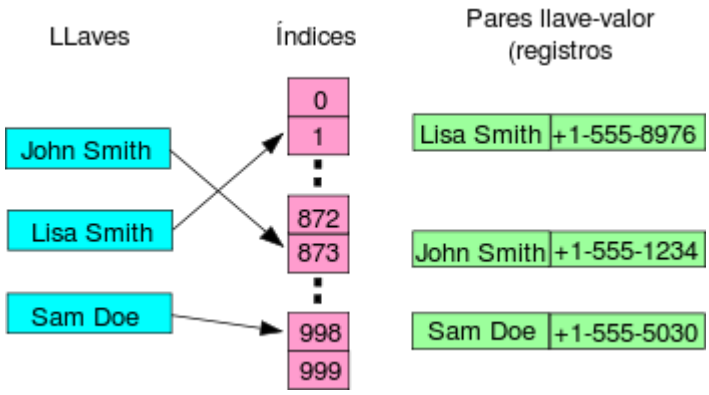
Para transformar cada subcadena de m caracteres en un entero lo que hacemos es representar los caracteres en una base B que en el planteamiento original coincide con el tamaño del alfabeto. Por tanto, el entero xi correspondiente a la subcadena de texto Ci...Ci+m-1 sería:

$$x_i=C_i\times B^{m-1}+C_{i-1}\times B^{m-2}+\ldots+C_{i+m-1}$$

podemos calcular el valor de xi+1 en función de xi:

$$x_{i+1}=x_i\times B-C_i\times B^{m}+C^{i+m}$$

Es decir, si la cadena es un número en base B, el nuevo valor será el resultado de multiplicar por la base el valor anterior eliminado el dígito de mayor peso (ya que no está en la cadena) y añadiendo como componente de menor peso el valor del nuevo símbolo.



Siguiendo este planteamiento, y dependiendo de la longitud del patrón, el xi podría superar el rango de enteros representable por el computador. Para que esto no suceda se usa la función módulo (resto de la división). Como la función módulo es asociativa, podemos calcular el xi+1 incrementalmente a partir de cada xi.

Desarrollo de la Práctica

Se propusieron la cadena y el patrón, y como número primo se puso 29:

```
int main ()
{
  char texto[] = "aaaabcaaaos";
  char patron[] = "aaa";
  int q = 29;
  buscar (patron, texto, q);
  return 0;
}
```

Resultado de la búsqueda:

```
Patron encontrado en la posicion: 0
Patron encontrado en la posicion: 1
Patron encontrado en la posicion: 6
```

Para el segundo programa cambié el incremento del contador que recorre la cadena, si encuentra la cadena avanza el equivalente del tamaño del patrón, y sino sólo avanza 1 caracter.

Prueba 1:

```
int main ()
{
  char texto[] = "aaaabcaaaos";
  char patron[] = "aaa";
  int q = 29;
  buscar (patron, texto, q);
  return 0;
}
```

```
Patron encontrado en la posicion: 0
Patron encontrado en la posicion: 6
```

Prueba 2:

```
int main ()
{
  char texto[] = "aaaabcaaaos";
  char patron[] = "aa";
  int q = 29;
  buscar (patron, texto, q);
  return 0;
}
```

```
Patron encontrado en la posicion: 0
Patron encontrado en la posicion: 2
Patron encontrado en la posicion: 6
```

Conclusión

El algoritmo Karp-Rabin es confuso al leerlo, pero al ponerlo en práctica resulta mucho más fácil de lo que parece, ya que se basa en una serie de operaciones y ciclos “for” que recorren las cadenas comparando hasta encontrar el patrón que se busca.