



Departamento de Ingeniería Informática
Universidad de Santiago de Chile

EXPERIENCIA 7 MÉTODOS DE PROGRAMACIÓN

AUTOR:
Diego Riquelme

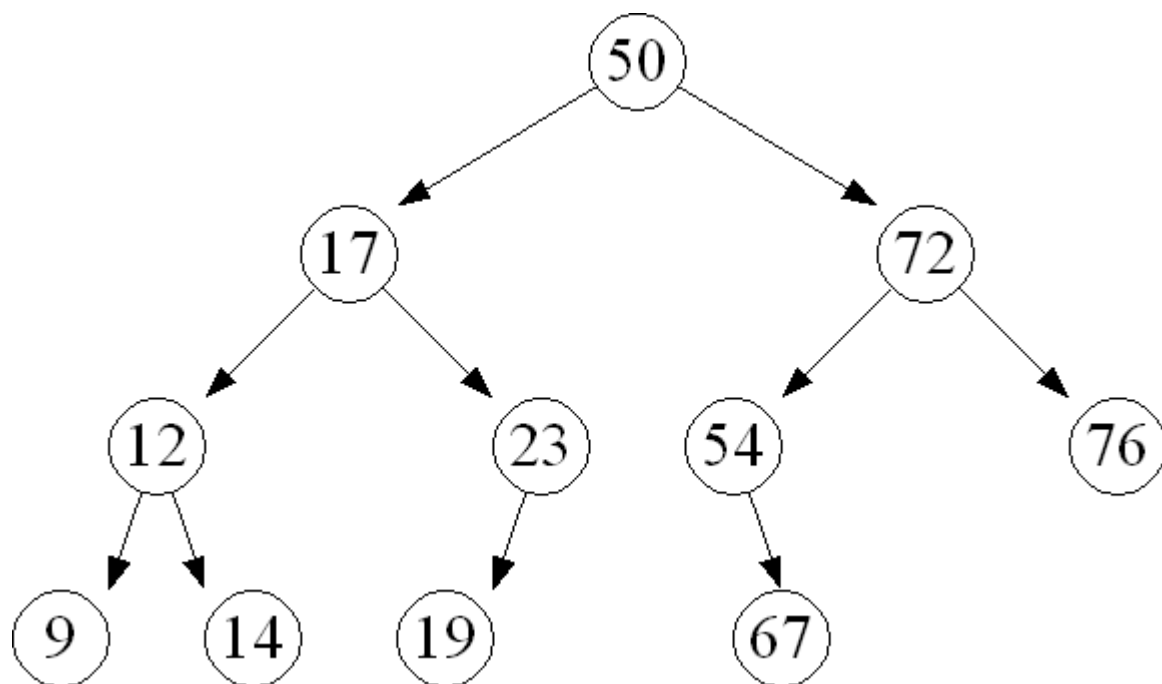
PROFESOR LABORATORIO:
Pablo Román

AYUDANTE:
Damián Guzman

Santiago, Chile - Otoño 20

Dentro de la informática, se pueden encontrar diversas estructuras de datos, tales como los arrays, pilas o registros, pero en esta ocasión, se trabajarán las estructuras arbóreas, las cuales son, *“Una configuración en árbol o topología en árbol o estructura en árbol es una caracterización física de un objeto y sus componentes, que por su configuración se asemeja o recuerda a un árbol, en el sentido que sus ramificaciones tienden a converger en un origen o raíz (por ejemplo el llamado árbol genealógico). Con este concepto se introduce por lo tanto las nociones de raíz y de descendencia. En informática es vulgarmente utilizada como (como estructura), junto a otras topologías en anillo o topologías en estrella. En programación informática, estas topologías son utilizadas como estructuras de datos para resolver problemas complejos donde por ejemplo debe aplicarse indexación.”*^[1]

A modo de ejemplo, tenemos el siguiente grafo:



Ahora bien, dentro del laboratorio de métodos de programación, nos piden implementar un algoritmo el cual, permita contar el numero que más se repite dentro de un grafo arbóreo como el anteriormente mostrado, pero con números totalmente aleatorios.

Si bien, parece una tarea sencilla, inmediatamente nos planteamos la pregunta, puedo contar en un ala del árbol el número que más veces se repite, pero ¿cómo sé con certeza que este será el número más repetido de la estructura completa?, para ello, se desarrolla el siguiente algoritmo, el cual nos permitirá buscar la moda dentro de todo el árbol, y para ello, empezamos a dividir nuestro problema principal en subproblemas.

Para comenzar, tenemos la certeza de poder encontrar la moda dentro del **ala izquierda**, es por ello, que procedemos a calcularla y a contar la cantidad de veces que aparece, luego de ello, verificamos si la moda calculada dentro de nuestra ala **izquierda** esta dentro del ala **derecha**, de ser así, procedemos a sumar en la cantidad de veces necesarias que esta tenga a nuestro contador anterior, sino, nuestra moda seria la calculada en el ala izquierda, y la cantidad de veces aparecida seria solo la que está en esta misma.

Una vez finalizando la tarea anterior, podemos proceder con el ala **derecha**, donde buscaremos la moda de igual manera y contar la cantidad de veces que aparece esta, luego de ello, preguntar si esta aparece en el ala **izquierda**, tal cual lo hicimos anteriormente, y si apareciera, volvemos a contar sobre el contador del ala derecha, si no, dejamos el contador tal cual, retornando la moda y el contador obtenido.

Finalmente, preguntamos cual es la moda que más se repite a nivel árbol, y dependiendo de esta, **será el número más repetido** en todo el grafo.

Si hacemos un paso a paso, tendríamos:

Ala Izquierda

- 1._Buscar moda ala izquierda
- 2._Verificar existencia de moda ala izquierda en ala derecha
- 3._Si existe, contar y sumar con el contador anterior
- 4._Si no, quedan los datos del punto 2

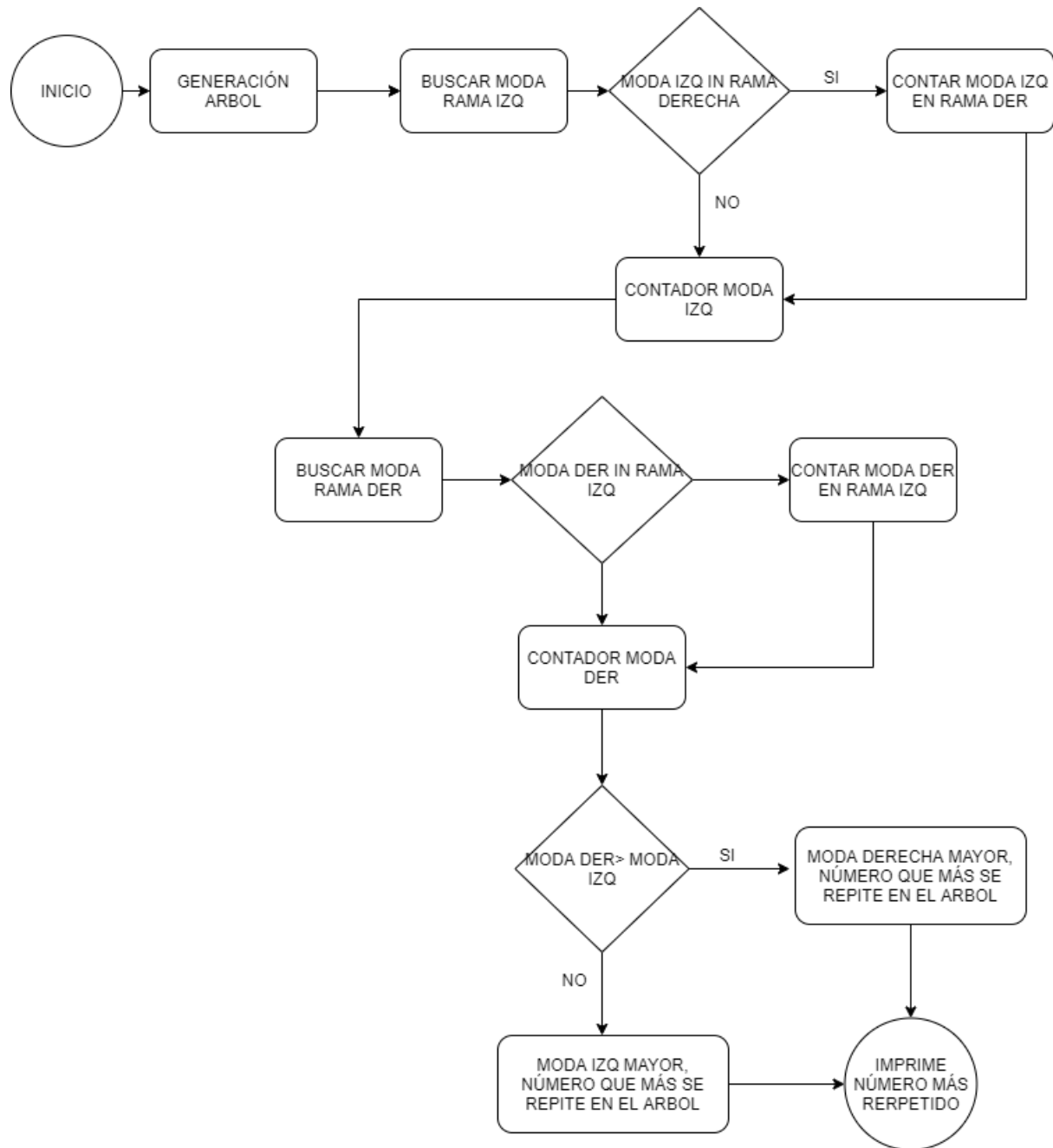
Ala Derecha

- 1._Buscar moda ala Derecha
- 2._Verificar existencia de moda ala derecha en ala izquierda
- 3._Si existe, contar y sumar con el contador anterior
- 4._Si no, quedan los datos del punto 2


Final

- 1._Comparar contadores de alas, el que más se repita será el del árbol.

Si vemos lo anteriormente descrito dentro de un diagrama de flujo, quedaría algo así:



Por otro lado, para corroborar que este algoritmo funcione, lo verificaremos a través de un programa implementado en C, esto solo para ver su función, ya que los demás métodos de programación serán aplicados en el código final de la experiencia 8, además cabe destacar que dentro del código a continuación no se trabajó con topografía arbórea, sino con arrays para poder simplificar el código y la lectura de este (*por otra parte, el trabajo con la estructura de control arbórea es parte de la experiencia 8*).

A screenshot of a code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The code is written in C and is a function named 'in' that checks if a number is present in an array. The code is as follows:

```
#include<stdio.h>
//preguntar si esta dentro de la lista
int in(int lista[],int numero,int longitud){
    for (int i = 0; i < longitud; i++){
        {
            if (numero==lista[i])
            {
                return 1;
            }
        }
    }
    return 0;
}
```

-Partimos nuestro código incluyendo la librería estándar de entradas y salidas, además de crear una función que nos responde si un número está dentro de un array que nos servirá más adelante

```
void main(){
    int n,s,repeticion=0,a;
    int ramaIzq[5];
    int ramaDer[5];
    int num;
    //ingreso array izq
    printf("ingreso rama izquierda: \n");
    for (int i = 0; i < 5; ++i)
    {
        num=0;
        scanf("%d",&num);
        ramaIzq[i]=num;
        printf("----\n");
    }

    //ingreso array der
    printf("ingreso rama derecha: \n");
    for (int i = 0; i < 5; ++i)
    {
        num=0;
        scanf("%d",&num);
        ramaDer[i]=num;
        printf("----\n");
    }
}
```

-Luego, creamos nuestra función main donde ingresaremos hasta 5 números al array ala derecha y al array ala izquierda.

```

//buscar moda ramaIzq
for(int i=0;i<5;i++){
    s=0;
    for(int j=0;j<5;j++){
        if(ramaIzq[i]==ramaIzq[j] && i!=j){
            s=s+1;
        }
    }
    if(s>=repetidor){
        repetidor=s;//guardamos el mayor por el momento
        a=i;
    }
}
int repIzq;
repIzq=ramaIzq[a];
//m tiene las repeticiones pero no se esta contando a si misma por eso m+1
printf("la moda izq es: %d y tiene %d repeticiones\n\n",ramaIzq[a],repetidor+1);
//verificar existencia en el array derecho
if (in(ramaDer,repIzq,5)==1){
    for (int i = 0; i < 5; ++i){
        if(repIzq==ramaDer[i]){
            repetidor=repetidor+1;
        }
    }
    printf("en el brazo der se repiten más %d!\n",ramaIzq[a]);
    printf("la moda izq es: %d y tiene %d repeticiones en total con ambos brazos\n\n",ramaIzq[a],repetidor+1);
}

```

-En este punto, empezamos con la búsqueda de la moda en el ala izquierda, y todo lo que ello conlleva explicado anteriormente.

```

//borrar datos

int m=0;
s=0;
int b=0;
//buscar moda ramaDer
for(int i=0;i<5;i++){
    s=0;
    for(int j=0;j<5;j++){
        if(ramaDer[i]==ramaDer[j] && i!=j){
            s=s+1;
        }
    }
    if(s>=m){
        m=s;//guardamos el mayor por el momento
        b=i;
    }
}
int repDer;
repDer=ramaDer[b];
printf("la moda der es: %d y tiene %d repeticiones\n",ramaDer[b],m+1);
//verificar existencia en el array izq
if (in(ramaIzq,repDer,5)==1){
    for (int i = 0; i < 5; ++i){
        if(repDer==ramaIzq[i]){
            m=m+1;
        }
    }
    printf("en el brazo izq se repiten más %d!\n",ramaDer[b]);
    printf("la moda izq es: %d y tiene %d repeticiones en total con ambos brazos\n\n",ramaDer[b],m+1);
}

```

-Ahora, realizamos la búsqueda dentro del ala derecha, con sus pasos pertinentes.

```

if (repetidor>m)
{
    printf("El numero que más se repite es: %d\n",ramaIzq[a]);
}
else{
    printf("El numero que más se repite es: %d\n",ramaDer[b]);
}
}

```

-Finalmente comparamos, y entregamos el resultado por pantalla al usuario.

Conclusión

A modo de síntesis, se pudo conocer de la existencia de las diferentes formas de estructuras de datos, más específicamente en las de topología arbórea, las cuales dentro de ramos como Análisis de algoritmo y estructura de datos, se revisan a profundidad, por otro lado, se pudo abstraer la información correspondiente para la creación de un algoritmo el cual pudiera resolver el problema planteado, pero no hay que dejar de lado que este proceso es meramente experimental por el momento y falta la respectiva evaluación por parte del docente o encargado de corrección.

Codigo anterior en GitHub: <https://github.com/driques/ExampleExp7>