MATRICES DISPERSAS - FORMATO ELLPACK

Cecilia Brusquetti
Patricia Fuster
Belén Mareco
Gricelda Valdez

1.Crear una matriz completa: al pasar de la representación a la forma matricial

Recibe como parámetros el puntero al primer nodo de la lista enlazada, que utilizamos para la representación ELLPACK, la cantidad de filas M de la matriz y la cantidad de columnas N. Recorremos con un for para ir imprimiendo la matriz, dependiendo de que si en la lista enlazada utilizada para la representación se encuentra dicha posición de fila, columna y elemento correspondiente, de lo contrario esto corresponde a un “0” en la matriz.
La función llamada matricial en la implementación, recorre toda la lista enlazada para así ir imprimiendo todos los elementos, en sus correspondientes posiciones.
Por lo tanto el costo de la función matricial, será 0(n^2)

2. Obtener representación

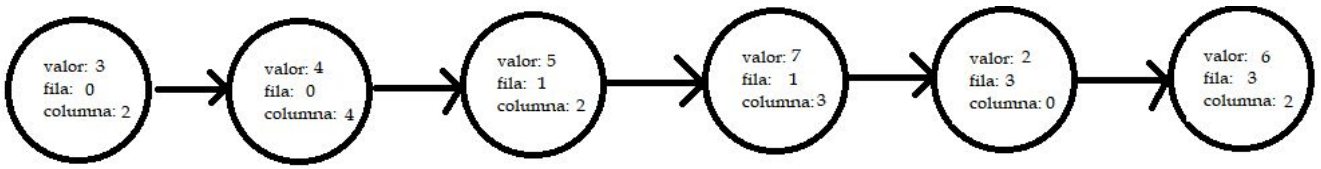
Dada una matriz cualquiera, de tamaño mxn, el procedimiento para conseguir su representación ELLPACK es el siguiente:

1. Se recorre la matriz dada para encontrar los elementos no nulos. Como esta es una operación costosa ($O(n^2)$), se realiza una sola vez, y se guardan los elementos relevantes en una lista enlazada que se crea dinámicamente. Cada inserción a la lista enlazada tiene un costo O(1). Cada nodo de la lista enlazada contiene el valor almacenado, la fila de la matriz dispersa y la columna de la matriz dispersa, además de un puntero al siguiente elemento de la lista.
2. Por otro lado, para generar la representación en el formato ELLPACK, es necesario tener un elemento mas de informacion: k, que contiene el máximo valor de elementos no nulos por fila de la matriz dispersa. Para encontrar este numero, se recorre solamente la lista enlazada ($O(n)$),contando cuántos elementos hay en cada fila y simplemente guardando cada vez que se alcanza un numero mayor.
3. Una vez que ya se tiene k, simplemente se imprime la lista enlazada en formato de dos matrices densas de mxk.

Un ejemplo podría ser la matriz:

0	0	3	0	4
0	0	5	7	0
0	0	0	0	0
0	2	6	0	0

Luego del primer paso, nos quedaría una lista enlazada de esta forma:



Como podemos ver, hay máximo 2 elementos por fila, por lo que k=2. Luego, las matrices densas de la representación son de tamaño mxk:

DATOS:			INDICES:	
	3	4	2	4
	5	7	2	3
	*	*	*	*
	2	6	0	2

3. Obtener elemento

Dada una representación y los parámetros de una posición, se debe retornar el elemento al cual pertenece dicha posición. Para ello se recorre primeramente las filas de la representación hasta encontrar el valor de la fila buscado. Luego se realiza exactamente lo mismo hasta encontrar el valor de la columna esperada. Al obtener el puntero asociado a dicha posición, se retorna finalmente el elemento buscado.

El costo de la función obtener elemento es del orden O(n), ya que en el peor de los casos se debe recorrer toda la lista enlazada en ambas direcciones (filas y columnas).

4. Obtener fila asociada

Dada una representación y una fila i, se imprime los valores asociados a la fila que corresponde a la matriz dispersa. La estrategia es la siguiente:

1. Declaramos un vector de enteros e inicializamos a cero todos sus elementos. La longitud del vector será la longitud de la columna de la matriz, pues la fila se mantiene fija y lo que va cambiando son los valores de las columnas. Este vector será el que contendrá a la fila asociada.
2. Recorremos la lista enlazada que guarda los datos útiles de la matriz dispersa, vamos preguntando si la fila es igual lista->pos_fila. Si es verdadero entonces en vector[lista->pos_columna] asignamos lista->valor; y actualizamos el puntero de la lista al siguiente. Si es falso simplemente actualizamos el puntero de la lista al siguiente.
3. El ítem 2 lo que hace es modificar los valores en el vector de acuerdo a la representación. Por lo tanto, al recorrer toda la lista, ya tenemos en el vector guardado los valores asociados a la fila i.
4. Finalmente imprimimos los valores del vector.
5. El costo de la función obtener_fila es de O(n), pues en el peor de los casos se tendrá que recorrer toda la lista enlazada.

5. Obtener columna asociada

Dada una representación y una columna j, se imprime los valores asociados a la columna que corresponde a la matriz dispersa. La estrategia es la siguiente:

1. Declaramos un vector de enteros e inicializamos a cero todos sus elementos. La longitud del vector será la longitud de la fila de la matriz, pues la columna se mantiene fija y lo que va cambiando son los valores de las filas. Este vector contendrá a la columna asociada.
2. Recorremos la lista enlazada que guarda los datos útiles de la matriz dispersa mediante un ciclo while, vamos preguntando si la columna es igual a lista->pos_columna. Si es verdadero entonces en vector[lista->pos_fila] asignamos lista->valor; y actualizamos el puntero de la lista al siguiente. Si es falso simplemente actualizamos el puntero de la lista al siguiente.
3. El ítem 2 lo que hace es modificar los valores en el vector de acuerdo a la representación. Por lo tanto, al recorrer toda la lista, ya tenemos en el vector guardado los valores asociados a la columna j.
4. Finalmente imprimimos los valores del vector.
5. El costo de la función obtener_columna es de O(n), pues en el peor de los casos se tendrá que recorrer toda la lista enlazada.

6.Modificar posición

Dada una representación, un posición i,j y un elemento, debe modificarse la representación para ingresar este elemento. La estrategia es la siguiente:

Contemplamos un caso base y caso general. Para el caso base (caso 1) el elemento que se quiere agregar y su posición ya están en la representación por lo que se debe sobreescribir el campo lista->valor= elemento. Para el caso general se contemplan tres subcasos, que son:

1. El elemento a agregar esta en una posición anterior al comienzo de la lista, por lo que se debe crear un nuevo nodo, y el puntero al comienzo de la lista debe apuntar a este nodo creado.
2. El elemento a agregar esta en una posición posterior al último elemento de la lista, por lo que se debe crear un nuevo nodo, y el puntero al siguiente de este nodo creado debe ser NULL, indicando que es el último elemento de la lista.
3. El elemento a agregar esta en una posición intermedia entre dos nodos de la lista, por lo que se debe crear un nuevo nodo, cuyo anterior a la lista debe apuntar a este, y a su vez este nodo debe apuntar al siguiente que había originalmente en la lista. Para recorrer la lista usamos dos variables auxiliares.
4. En el programa principal una vez llamada a esta función, se llama inmediatamente a la función que imprime la representación, pudiendo observar los cambios en la lista y por tanto en la representación.
5. Esta función tiene un costo de O(n) puesto que tiene que recorrer toda la lista en el peor de los casos. Y en el mejor de los casos puede tener un costo de O(1) puesto que en el ciclo while una vez agregado el elemento, inmediatamente se fuerza la salida del ciclo.

7. Sumar matrices

Para realizar una suma entre dos representaciones se necesitan como parámetros los siguientes datos: el puntero al inicio de la lista enlazada, que contiene la representación de la matriz dispersa, y la cantidad de filas que esta matriz posee. Al realizar la operación se obtiene una representación resultante, con su lista de datos e índices correspondientes. Primeramente, se compara y se obtiene el mayor valor del parámetro k entre ambas representaciones, para que sirva como límite al recorrer las posiciones de las representaciones.

Para realizar la suma entre ambas representaciones se debe tener en cuenta que coincidan las posiciones, es decir la fila y la columna de cada posición deben ser iguales. Para eso, al momento de recorrer cada lista enlazada se hacen ciertas consultas antes de realizar dicha operación entre ambas posiciones:

1. Si las filas y las columnas de ambas posiciones coinciden, se realiza con éxito la operación y se coloca el índice adecuado en la representación resultante.
2. Si las filas coinciden pero las columnas no, se copia el valor de la menor columna. Por ejemplo, si ambas posiciones están en la misma fila, pero uno posee el valor de la columna c=2 y la otra c=3, se copia el valor de la columna c=2 , para que puedan estar en orden.
3. Si las filas no coinciden, se copia el valor de la posición con el valor de la fila menor.

En el caso de que se haya copiado un valor con la columna o fila menor de una representación, la posición de la otra representación se sigue comparando por si se pueda obtener un valor que coincida con los parámetros de su posición.

Finalmente al realizar todas las consideraciones se llama a la función crear_nodo, el cual se encarga de crear un nuevo nodo para la representación resultante.

El costo de dicha función es del orden $O(n^2)$.

8. Matriz transpuesta

Para poder retornar la representación de la matriz transpuesta se debe recibir la lista enlazada que contiene la representación de la matriz dispersa. Luego, se recorre la lista y se crea una nueva lista en la que se guardan los mismos elementos pero con los elementos de las filas y columnas invertidos, es decir, por ejemplo, si en la matriz original el valor 3 se encontraba en la fila 0 y columna 2, al almacenarlo en la transpuesta se convierte en el valor 3 de la fila 2 y columna 0. Posteriormente, se realiza el mismo procedimiento de encontrar la k de la matriz y se puede imprimir su representación. Como se mencionó anteriormente, recorrer la lista enlazada tiene un costo O(n), e insertar un elemento a la lista nueva tiene un costo O(1) por vez.

Conclusiones:

Para finalizar, es una implementación bastante sencilla pero eficiente. Probablemente no logra los mejores costos de ejecución, pero estos se encuentran dentro de lo razonable ya que no son excesivos.

BIBLIOGRAFÍA - WEBGRAFÍA:

https://www.geeksforgeeks.org/merge-sort-for-linked-list/ (Inspiracion para mergesort)
Introduction to Algorithms- Third Edition (Cormen et al.) (Costos para algoritmos)