Representación de grafos y digrafos

Algoritmos y Estructuras de Datos III

8 de abril de 2022

Grafos y digrafos

Definición

Un grafo es un par (V, E) con V un conjunto de nodos y E conjunto de aristas de la forma $\{u, v\}$ con $u, v \in V$.

Definición

Un digrafo es un par (V, E) con V un conjunto de nodos y E conjunto de aristas de la forma (u, v) con $u, v \in V$.

Operaciones

Algunas operaciones que nos gustaría poder calcular:

- adyacentes(u, v): decide si $uv \in E$.
- vecinos(v): conjunto de vértices adyacentes a v.
- agregarVertice(v), removerVertice(v).
- agregarArista(u, v), removerArista(v).

Sobre digrafos, en lugar de adyacentes y vecinos, tenemos hayArco(u, v), vecinos$_{in}$(v) y vecinos$_{out}$(v).

Representaciones: versión abstracta

¿Cómo podemos representar un grafo?

- Conjunto de aristas: almacenar el E de la definición de grafo. Cada arista es un par de nodos.
- Diccionario: a cada vértice se le asocia $N(v)$, su conjunto de vértices vecinos.

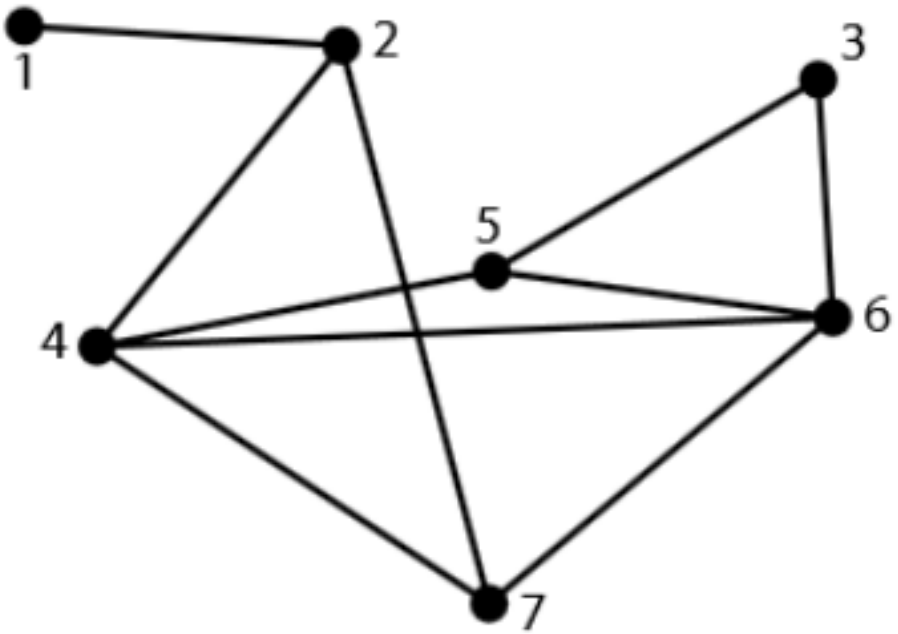
Representaciones: más concretamente

- Conjunto de aristas: Lo podemos representar con una lista enlazada, con un vector (ordenado o no), con un AVL, etc.
- Diccionario(vértices, vecindarios): Puede ser un AVL, una tabla de hash, un vector, etc. A su vez, los vecindarios son conjuntos, pueden ser listas, vectores (ordenados o no), tablas de hash, etc.

Representaciones: Lo más común

- Conjunto de aristas como una secuencia (lista de incidencia).
- Matriz de adyacencia: El diccionario es un vector y los vecindarios son vectores de tamaño n. M_{ij} es 1 si i y j son adyacentes y 0 si no.
- Lista de adyacencia: El diccionario es un vector y los vecindarios son vectores de tamaño $d(v)$, la lista de vecinos.

Representaciones: Ejemplo



Conjunto de aristas:
$\{(1,2),(2,4),(2,7),(4,5),$
$(4,7),(5,3),(5,6),(6,7)\}$

Matriz de adyacencia:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Lista de adyacencia:

$L_1 : 2$
$L_2 : 1 \rightarrow 4 \rightarrow 7$
$L_3 : 5 \rightarrow 6$
$L_4 : 2 \rightarrow 5 \rightarrow 6 \rightarrow 7$
$L_5 : 3 \rightarrow 4 \rightarrow 6$
$L_6 : 3 \rightarrow 4 \rightarrow 5 \rightarrow 7$
$L_7 : 2 \rightarrow 4 \rightarrow 6$

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción			

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$		

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes			

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$		

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos			

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$		

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista			

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$		

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista			

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$		

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	$O(d(u)+d(v))$
agregarVértice			

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	$O(d(u)+d(v))$
agregarVértice	$O(1)$		

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	$O(d(u)+d(v))$
agregarVértice	$O(1)$	$O(n)$	

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	$O(d(u)+d(v))$
agregarVértice	$O(1)$	$O(n)$	$O(1)$
removerVértice			

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	$O(d(u)+d(v))$
agregarVértice	$O(1)$	$O(n)$	$O(1)$
removerVértice	$O(m)$		

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	$O(d(u)+d(v))$
agregarVértice	$O(1)$	$O(n)$	$O(1)$
removerVértice	$O(m)$	$O(n^2)$	

Complejidades

	lista de aristas	matriz de ady.	listas de ady.
construcción	$O(m)$	$O(n^2)$	$O(n+m)$
adyacentes	$O(m)$	$O(1)$	$O(d(v))$
vecinos	$O(m)$	$O(n)$	$O(1)$
agregarArista	$O(1)/O(m)$	$O(1)$	$O(1)/O(d(v))$
removerArista	$O(1)/O(m)$	$O(1)$	$O(d(u)+d(v))$
agregarVértice	$O(1)$	$O(n)$	$O(1)$
removerVértice	$O(m)$	$O(n^2)$	$O(n+m)$

Algunas ideas sobre la construcción

Observación

La construcción de la lista de aristas y la lista de adyacencias es lineal. La matriz de adyacencia no.
$\rightarrow$ Podemos suponer que arrancamos con cualquier construcción que tenga costo lineal, sin costo.

Ejemplo

- Podemos suponer que tenemos listas de adyacencia ordenadas según $d(v)$ o según índice.
- Podemos suponer que las entradas en las listas de adyacencia están asociadas a sus entradas simétricas con punteros.
- etc.

Generalmente el input es conjunto de aristas y lo primero que hacemos es transformar a la representación que necesitemos.

Ejercicio 1

Amigo de todos

Dada la matriz de adyacencia de un grafo, queremos implementar la operación amigoDeTodos(V') que, dado un conjunto de vértices V' del grafo, decide si alguno de esos vértices tiene grado $|V'| - 1$ en el subgrafo inducido por V'.

Amigo de todos: Solución 1

- ¿Qué es lo más directo que podemos hacer?

Amigo de todos: Solución 1

- ¿Qué es lo más directo que podemos hacer?
- Si V' es $i_1, \ldots, i_n$, podemos recorrer por cada vértice esos índices en su fila y ver que sean todos 1.

Amigo de todos: Solución 1

- ¿Qué es lo más directo que podemos hacer?
- Si V' es $i_1, \ldots, i_n$, podemos recorrer por cada vértice esos índices en su fila y ver que sean todos 1.
- Complejidad?

Amigo de todos: Solución 1

- ¿Qué es lo más directo que podemos hacer?
- Si V' es $i_1, \ldots, i_n$, podemos recorrer por cada vértice esos índices en su fila y ver que sean todos 1.
- Complejidad? $O(|V'|^2)$

Amigo de todos: Solución 2

- ¿Cómo podemos mejorar un poco esa solución?

Amigo de todos: Solución 2

- ¿Cómo podemos mejorar un poco esa solución?
- Si encontramos un 0 cortamos y pasamos a la siguiente fila.

Amigo de todos: Solución 2

- ¿Cómo podemos mejorar un poco esa solución?
- Si encontramos un 0 cortamos y pasamos a la siguiente fila.
- Complejidad?

Amigo de todos: Solución 2

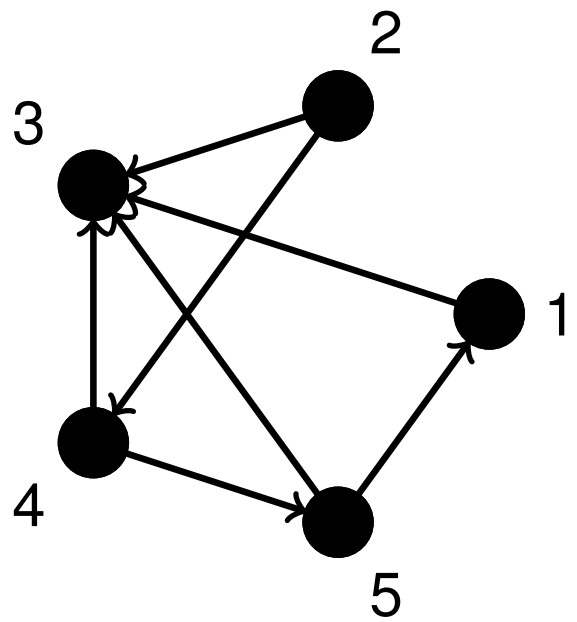
- ¿Cómo podemos mejorar un poco esa solución?
- Si encontramos un 0 cortamos y pasamos a la siguiente fila.
- Complejidad? $O(m'_V)$, la cantidad de aristas del grafo inducido por V'.

Ejercicio 2

Sumidero universal

Dado un digrafo $G = (V, E)$, queremos decidir si existe un vértice para el que todos los otros vértices son incidentes, pero que no incide sobre ningún vértice. ¿Qué estructura nos gustaría tener para hacer esto?

Ejemplo:



Sumidero universal: Matriz de adyacencias

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- ¿Cuál es la solución más directa?

Sumidero universal: Matriz de adyacencias

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- ¿Cuál es la solución más directa?
- ¿Podemos hacer algo mejor que $O(n^2)$?

Sumidero universal: Matriz de adyacencias

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- ¿Cuál es la solución más directa?
- ¿Podemos hacer algo mejor que $O(n^2)$?
- La idea del ejercicio anterior de cortar antes no nos sirve... Sigue siendo $O(n^2)$.

Sumidero universal: Matriz de adyacencias

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- Observación clave: Si en M_{ij} hay un 1, podemos descartar a i. Si hay un 0, podemos descartar a j.

Sumidero universal: Matriz de adyacencias

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- Observación clave: Si en M_{ij} hay un 1, podemos descartar a i. Si hay un 0, podemos descartar a j.
- ¿Cómo usamos esto?

Sumidero universal: Matriz de adyacencias

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

- Observación clave: Si en M_{ij} hay un 1, podemos descartar a i. Si hay un 0, podemos descartar a j.
- ¿Cómo usamos esto?
- Empezamos en $i = 1, j = 2$. En cada paso descartamos uno de los dos y reemplazamos el otro por el siguiente nodo. Cuando llegamos a n cortamos y nos queda un solo candidato. Luego chequeamos si es sumidero universal en $O(n)$. Complejidad total: $O(n)$.

Sumidero universal: Implementación con matriz de adyacencias

```
bool sumideroUniversal(vector<vector<bool> >& m){
    int n = m.size();
    int i = 0;
    int j = 1;
    while(i < n && j < n){
        if(m[i][j]){ // i → j, descartamos i
            i = max(i, j)+1;
        } else { // no hay arista, descartamos j
            j = max(i, j)+1;
        }
    }
    int cand = min(i, j);
    for(int k = 0; k < n; k++){
        if(k != cand && (!m[k][cand] || m[cand][k])) return false;
    }
    return true;
}
```

Sumidero universal: Lista de adyacencias

- Ahora supongamos que tenemos la lista de adyacencias. ¿Cómo lo podemos resolver?

Sumidero universal: Lista de adyacencias

- Ahora supongamos que tenemos la lista de adyacencias. ¿Cómo lo podemos resolver?
- Tenemos un solo candidato, el que tenga lista de tamaño 0. (lo encontramos en $O(n)$, genial)
- Tenemos que ver que le lleguen arcos de todos los otros vértices.

Sumidero universal: Lista de adyacencias

- Ahora supongamos que tenemos la lista de adyacencias. ¿Cómo lo podemos resolver?
- Tenemos un solo candidato, el que tenga lista de tamaño 0. (lo encontramos en $O(n)$, genial)
- Tenemos que ver que le lleguen arcos de todos los otros vértices.
- Complejidad: $O(m+n)$. Nos conviene la matriz.

Sumidero universal: Lista de adyacencias

- Ahora supongamos que tenemos la lista de adyacencias. ¿Cómo lo podemos resolver?
- Tenemos un solo candidato, el que tenga lista de tamaño 0. (lo encontramos en $O(n)$, genial)
- Tenemos que ver que le lleguen arcos de todos los otros vértices.
- Complejidad: $O(m + n)$. Nos conviene la matriz.
- ¿Qué pasa si además tenemos una lista de adyacencias invertida? Es decir, para cada nodo tenemos la lista de nodos que llegan a él.

Sumidero universal: Lista de adyacencias

- Ahora supongamos que tenemos la lista de adyacencias. ¿Cómo lo podemos resolver?
- Tenemos un solo candidato, el que tenga lista de tamaño 0. (lo encontramos en $O(n)$, genial)
- Tenemos que ver que le lleguen arcos de todos los otros vértices.
- Complejidad: $O(m+n)$. Nos conviene la matriz.
- ¿Qué pasa si además tenemos una lista de adyacencias invertida? Es decir, para cada nodo tenemos la lista de nodos que llegan a él.
- Buscamos el que tenga $d_{in} = n$ y $d_{out} = 0$. Con esto nos queda $O(n)$, y mucho más fácil!