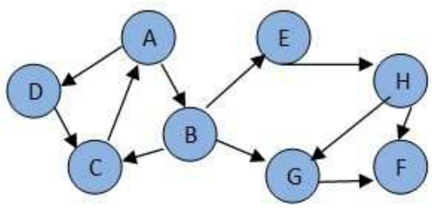
Analisi e progettazione di algoritmi

(III anno Laurea Triennale - a.a. 2023/24)

Prova scritta 17 giugno 2024

Esercizio 1 Si consideri il seguente grafo:



1. Si esegua la visita BFS iterativa, mostrando a ogni passo il nodo estratto, la coda restante e l'albero BFS parziale.
2. Si esegua la visita DFS iterativa, mostrando a ogni passo il nodo estratto, la pila restante e l'albero DFS parziale, evidenziandone la parte definitiva.
3. Si mostri il grafo quoziente.

In tutti i casi in cui si deve scegliere un nodo, si deve ottenere nella visita l'ordine alfabetico.

Soluzione (Data in formato non grafico per mia comodità.)

1.

nodo estratto	coda	albero parziale
	A	
A	$\cancel{A}$ B D	(A,B), (A,D)
B	$\cancel{A}$ $\cancel{B}$ D C E G	(A,B), (A,D), (B,C), (B,E), (B,G)
D	$\cancel{A}$ $\cancel{B}$ $\cancel{D}$ C E G	(A,B), (A,D), (B,C), (B,E), (B,G)
C	$\cancel{A}$ $\cancel{B}$ $\cancel{D}$ $\cancel{C}$ E G	(A,B), (A,D), (B,C), (B,E), (B,G)
E	$\cancel{A}$ $\cancel{B}$ $\cancel{D}$ $\cancel{C}$ $\cancel{E}$ G H	(A,B), (A,D), (B,C), (B,E), (B,G), (E,H)
G	$\cancel{A}$ $\cancel{B}$ $\cancel{D}$ $\cancel{C}$ $\cancel{E}$ $\cancel{G}$ H F	(A,B), (A,D), (B,C), (B,E), (B,G), (E,H), (G,F)
H	$\cancel{A}$ $\cancel{B}$ $\cancel{D}$ $\cancel{C}$ $\cancel{E}$ $\cancel{G}$ $\cancel{H}$ F	(A,B), (A,D), (B,C), (B,E), (B,G), (E,H), (G,F)
F	$\cancel{A}$ $\cancel{B}$ $\cancel{D}$ $\cancel{C}$ $\cancel{E}$ $\cancel{G}$ $\cancel{H}$ $\cancel{F}$	(A,B), (A,D), (B,C), (B,E), (B,G), (E,H), (G,F)

2.

nodo estratto	pila	albero parziale
	A	
A	~~A~~ B D	(A,B), (A,D)
B	C E G ~~A~~ ~~B~~ D	**(A,B)**, (A,D), (B,C), (B,E), (B,G)
C	~~C~~ E G ~~A~~ ~~B~~ D	**(A,B)**, (A,D), **(B,C)**, (B,E), (B,G)
E	H ~~C~~ ~~E~~ G ~~A~~ ~~B~~ D	**(A,B)**, (A,D), **(B,C)**, **(B,E)**, (B,G),(E,H)
H	F G ~~H~~ ~~C~~ ~~E~~ G ~~A~~ ~~B~~ D	**(A,B)**, (A,D), **(B,C)**, **(B,E)**, **(E,H)**, (H,F), (H,G)
F	~~F~~ G ~~H~~ ~~C~~ ~~E~~ G ~~A~~ ~~B~~ D	**(A,B)**, (A,D), **(B,C)**, **(B,E)**, **(E,H)**, **(H,F)**, (H,G)
G	~~F~~ ~~G~~ ~~H~~ ~~C~~ ~~E~~ G ~~A~~ ~~B~~ D	**(A,B)**, (A,D), **(B,C)**, **(B,E)**, **(E,H)**, **(H,F)**, **(H,G)**
G	~~F~~ ~~G~~ ~~H~~ ~~C~~ ~~E~~ ~~G~~ ~~A~~ ~~B~~ D	**(A,B)**, (A,D), **(B,C)**, **(B,E)**, **(E,H)**, **(H,F)**, **(H,G)**
D	~~F~~ ~~G~~ ~~H~~ ~~C~~ ~~E~~ ~~G~~ ~~A~~ ~~B~~ ~~D~~	**(A,B)**, **(A,D)**, **(B,C)**, **(B,E)**, **(E,H)**, **(H,F)**, **(H,G)**

3. Le componenti fortemente connesse sono $ABCD$, E, F, G, H, e nel grafo quoziente ci sono gli archi $(ABCD, E), (ABCD, G), (H, G), (H, F), (G, F)$.

Esercizio 2 Si consideri il problema della ricerca di un elemento x in una matrice $M[0..n-1, 0..n-1]$ nella quale *ogni riga* e *ogni colonna* è ordinata in modo crescente.

1. L'algoritmo "stupido" che esamina tutta la matrice ha ovviamente complessità (del caso peggiore) $O(n^2)$. Descrivere il caso migliore e il caso peggiore.

2. Dare un algoritmo più efficiente utilizzando la tecnica divide et impera. Suggerimento: si dia un algoritmo che ricerca x in una sottomatrice, confrontandolo con l'elemento in alto a destra (quindi, per la chiamata iniziale, con $M[0, n-1]$).

3. Giustificare la correttezza (o almeno la terminazione) dell'algoritmo proposto, indicando la tecnica utilizzata.

4. Analizzare il costo computazionale dell'algoritmo proposto.

Soluzione

1. Il caso migliore si ha quando l'elemento si trova nel primo elemento esaminato (tipicamente, $M[0, 0]$), il caso peggiore quando non compare nella matrice.

2. Un algoritmo divide et impera che decide se un elemento compare nella matrice è il seguente:

```
search(x,M)
  search(x,M,0,n-1)

search(x,M,i,j) // cerca x nella sottomatrice M[i..n-1,0..j]
  if (i ≤ n && 0 ≤ j) case
    M[i,j]) < k : return search(x,M,i+1,j)
    M[i,j]=k: return true
    M[i,j]) > k : return search(x,M,i,j-1)
  return false
```

3. La chiamata `search(x,M,i,j)` cerca x nella sottomatrice $M[i..n-1, 0..j]$. Possiamo provare la correttezza ragionando per induzione forte, infatti:

   - se $i > n-1$ oppure $j < 0$ la sottomatrice è vuota, e l'algoritmo termina correttamente restituendo `false`
   - altrimenti, le chiamate ricorsive sono su matrici strettamente più piccole, ossia $M[i+1..n-1, 0..j]$ oppure $M[i..n-1, 0..j-1]$; nel primo caso si elimina una riga in quanto questa contiene necessariamente elementi più piccoli di x, nel secondo caso si elimina una colonna in quanto questa contiene necessariamente elementi più grandi di x.

4. Come osservato sopra, a ogni chiamata ricorsiva si elimina una riga o una colonna, quindi la complessità è lineare. Utilizzando le relazioni di ricorrenza, e indicando con n la somma delle dimensioni della matrice, si ha infatti:

   $T(0) = 1$
   $T(n) = 1 + T(n-1)$, per $n > 0$.

   e utilizzando la tecnica per sostituzioni successive si trova la somma $1 + \ldots + 1$ (n volte), quindi $T(n) = O(n)$.

Esercizio 3 La matrice dei pagamenti di un gioco tra Roberta e Carlo è

$$\mathbf{M} = \begin{pmatrix} 5 & 2 \\ 1 & 4 \end{pmatrix}$$

Dimostra che $V_R < V_C$, determina le strategie miste ottimali di Roberta e Carlo e il valore del gioco.