Heap di Fibonacci
Lezione n°4

Prof.ssa Rossella Petreschi

Lezione del 17 /10/2011 del

Corso di Algoritmi e Strutture Dati

Riferimenti:

Capitolo 21 del testo Cormen, Leiserson, Rivest

“Introduzione agli algoritmi”Edizioni: Jackson Libri

Capitolo 4 del testo Ausiello, Petreschi

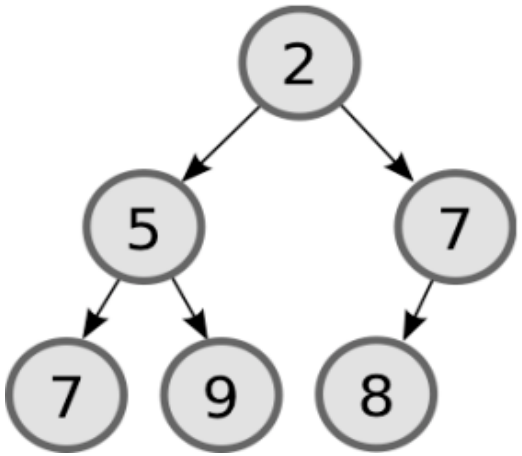
“L’informatica invisibile…”Edizioni: Mondadori/Sapienza

Heap

La struttura dati Heap (minimo) è un albero binario di n elementi, completo o quasi completo, tale che il valore associato ad ogni nodo sia maggiore o uguale a quello associato al padre. La radice contiene il valore minimo(in modo analogo si parla di heap massimo). L'heap è efficientemente rappresentato con un vettore e

$\forall$ i , p(i) in posizione $\lfloor i/2 \rfloor$, fs(i) e fd(i) in posizione 2i e 2i+1

2	5	7	7	9	8
1	2	3	4	5	6



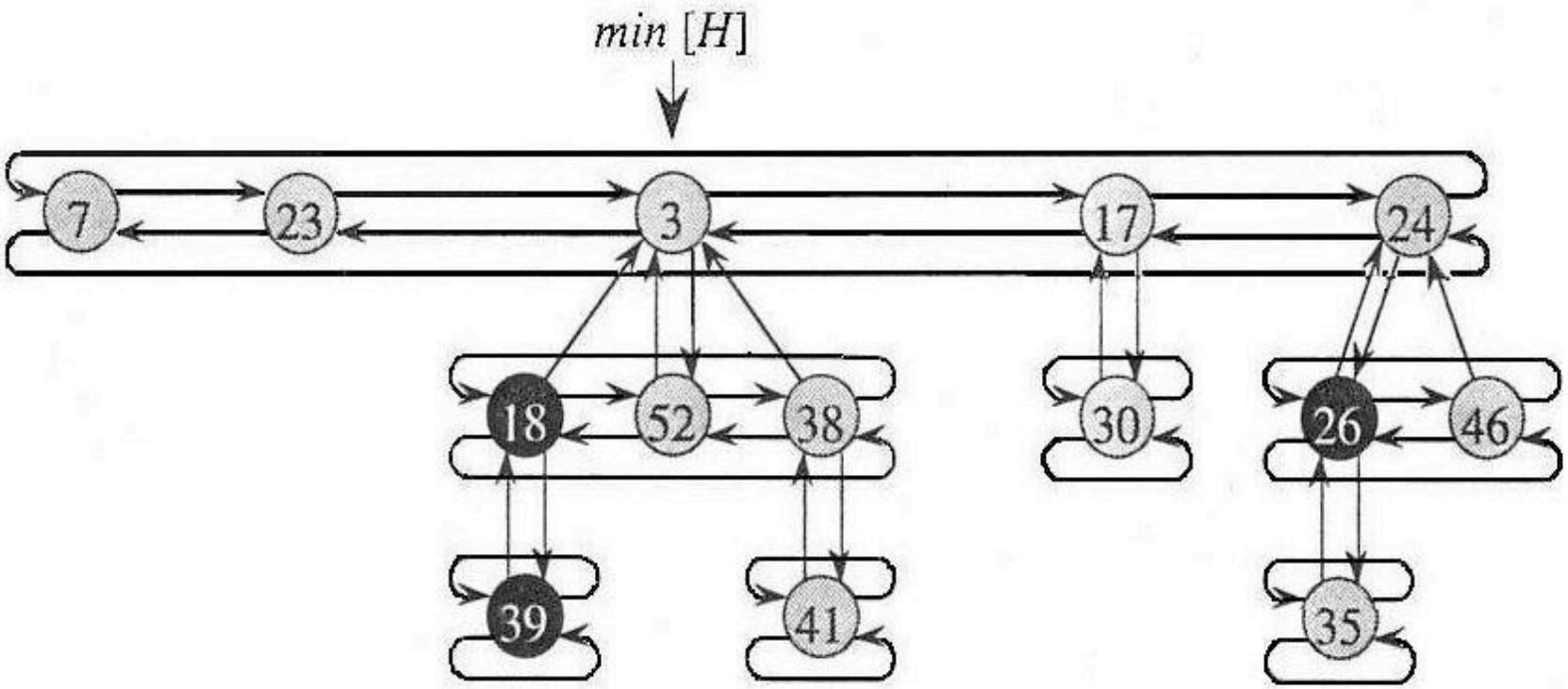
Heap di Fibonacci: definizione

Un Heap di Fibonacci è un insieme non ordinato di alberi con la seguente struttura:

- le radici degli alberi dell'heap, così come i figli di ogni singolo nodo, sono fra loro collegate da una lista doppiamente puntata;
- ogni nodo v contiene
  - un puntatore al padre;
  - un puntatore ad uno dei figli;
  - il numero dei figli;
  - un valore inferiore a quello associato ai figli (per ogni albero il valor minimo è associato alla radice);
- esiste un puntatore alla radice contenente il valore minimo di tutto l'heap.

Infine ad ogni nodo v è associato un booleano. Il booleano Vero indica che v è marcato, dato che ha perso un figlio durante una delle operazioni sull'heap.

Heap di Fibonacci: immagine



Funzione potenziale

$$\Phi(F) = t(F) + 2m(F)$$

t(F) numero degli alberi

m(F) numero dei nodi marcati

Operazione di inserimento

Heap: il nuovo nodo con valore x è inserito come foglia più a destra nell'ultimo livello dell'Heap preesistente. Viene poi riaggiustato l'Heap scambiando il nuovo nodo con il padre se il valore ad esso associato è maggiore di x. Il procedimento può iterare fino a raggiungere la radice. O(logn).

Heap di Fibonacci: si crea un semplice Heap di Fibonacci costituito dal solo nodo di valore x. Si uniscono le liste delle radici del nuovo Heap e di quello preesistente. Si aggiorna il valore del numero dei nodi nello heap e, se occorre,il puntatore al minimo.

$\chi_i = c_i + \Phi(D_i) - \Phi(D_{i-1}) = \Theta(1) + (t(F_i) + 2m(F_i)) - (t(F_{i-1}) + 2m(F_{i-1})) =$
$\Theta(1) + ((t(F_{i-1})+1) + 2m(F_{i-1})) - (t(F_{i-1}) + 2m(F_{i-1})) = \Theta(1) + 1 = \Theta(1)$

Trova il minimo è costante sia nell'Heap che nell'Heap di Fibonacci

Estrazione del minimo dall'Heap

Estrai il minimo dall'Heap

Si estrae il minimo e poi si sostituisce alla radice il valore della foglia più a destra nell'ultimo livello dell'Heap. Si elimina la foglia, riducendo di uno il numero di nodi nell'Heap. Si riaggiusta l'Heap scambiando il nuovo nodo radice con il figlio di valore minore, sempre che non sia uguale a quello associato alla radice. Il procedimento può iterare fino a raggiungere una foglia. O(logn).

Estrazione del minimo dall'Heap di Fibonacci (1)

•Si portano i figli del nodo minimo al livello delle radici.

•Si elimina la radice minimo e il puntatore principale dell'Heap punta ad un nuovo nodo radice momentaneamente.

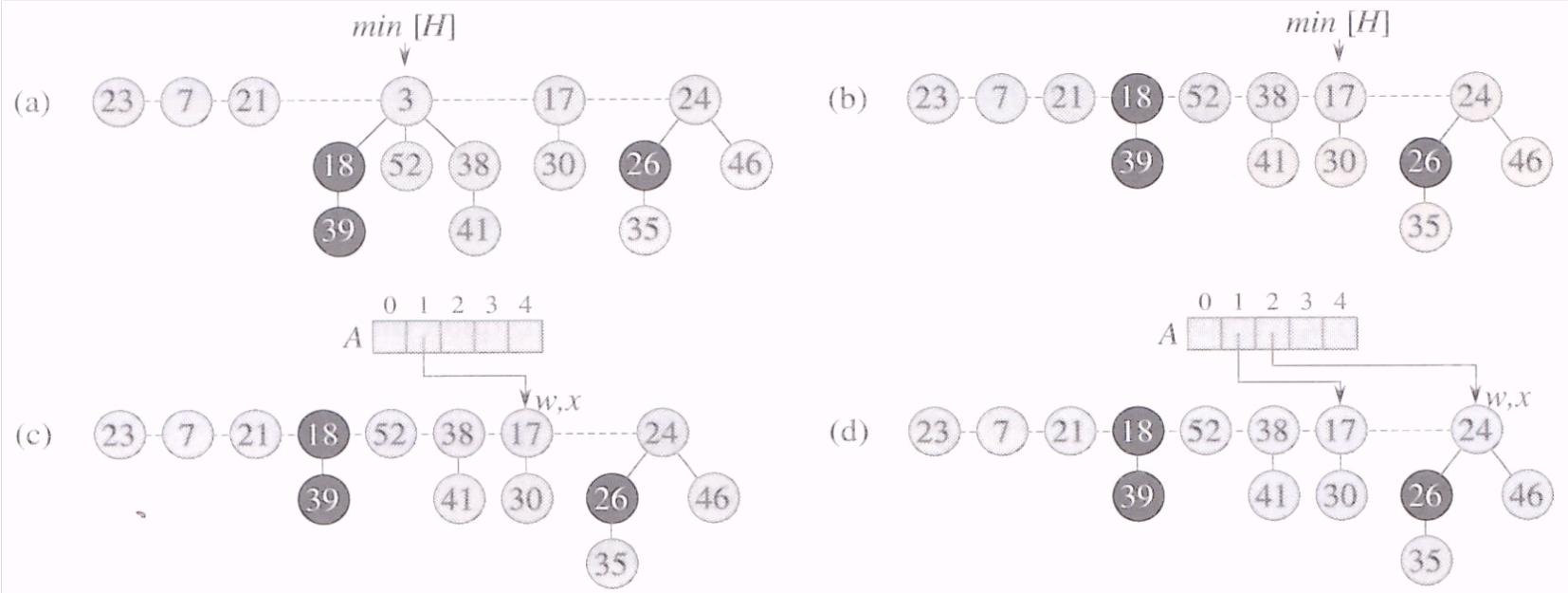
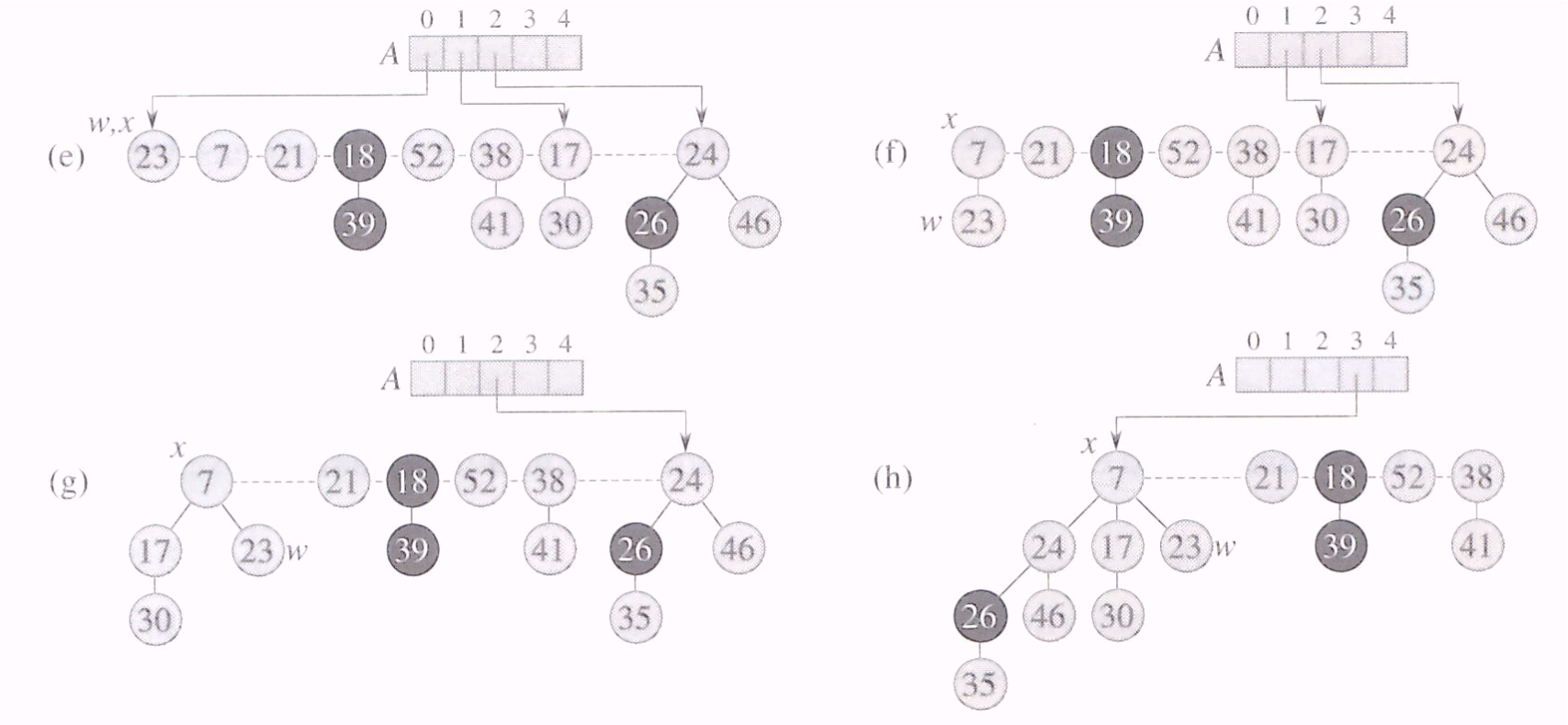
•Poi riaggiusta l'Heap con il seguente passo da ripetere fino a quando tutte le radici non avranno grado differente

- *siano x e y due radici con lo stesso numero di figli, Val(y)> Val (x), y diviene figlio di x e viene eliminata dalla lista delle radici*

(si incrementa il grado di x, mentre si analizzano le radici si ripristina il puntatore principale)

$$\chi_i = c_i + \Phi(D_i) - \Phi(D_{i-1}) = O(d(n)+ t(F_{i-1})) + ((d(n)+1) + 2m(F_{i-1})) - (t(F_{i-1}) + 2m(F_{i-1})) = O(d(n))+ O\ (t(F_{i-1})) + d(n) + 1 - t(F_{i-1}) = O(d(n))$$

Estrazione del minimo dall'Heap di Fibonacci(2)



Estrazione del minimo dall'Heap di Fibonacci (3)