

Fondamenti di Informatica II

Parte II: Algoritmi e strutture dati (Prof. Becchetti)

Diario delle lezioni - AA 2019/2020

Ricorsione

1. Esempi introduttivi
2. Ricerca binaria in un array e sua analisi
3. Calcolo della somma degli elementi in un array e sua analisi
4. Inversione di un array e sua analisi
5. Costi della ricorsione: elevamento a potenza e sua analisi
6. Ricorsione in coda e sua eliminazione
7. Ricorsione multipla
8. Calcolo dei numeri di Fibonacci ed efficienza: implementazioni con ricorsione doppia, ricorsione lineare e iterativa

Riferimenti: libro di testo, Capitolo 5 (tutto)

Code di priorità e heap

1. La coda di priorità come tipo di dato astratto: coppie (chiave, valore) e operazioni di base
2. Implementazione di code di priorità mediante liste
 1. Implementazione con liste ordinate e non ordinate
 2. Costo delle operazioni di base: vantaggi e svantaggi
3. La struttura dati heap
4. Implementazione di code di priorità con heap e sua analisi
5. Costruzione bottom-up di un heap (heapify) e sua analisi
6. Ordinamento con code di priorità e relativi costi: Selection sort, Insertion sort, Heap sort
7. Code di priorità con entry consapevoli della propria posizione: code prioritarie e heap flessibili

Riferimenti: libro di testo, Capitolo 9, dispense e appunti del docente disponibili sul sito del corso

Algoritmi di ordinamento

1. Ordinamento usando il metodo Dividi-e-conquista: Merge sort
 1. Definizione ricorsiva del Merge sort e analisi: correttezza e tempo di esecuzione
 2. Merge sort iterativo
2. Algoritmo di ordinamento Quick sort
 1. Algoritmo e analisi
 2. Scelta casuale del pivot e cenni alla sua analisi
 3. Ulteriori ottimizzazioni
3. Limite inferiore alla complessità dell'ordinamento basato sul confronto
4. Ordinamento in tempo lineare: bucket sort, ordinamento stabile, radix sort

5. Confronto tra algoritmi di ordinamento
6. Il problema della selezione: quick select probabilistico e sua analisi

Riferimenti: libro di testo, Capitolo 12. Appunti del docente sul Merge sort iterativo

Mappe e insiemi

1. Mappe, insiemi e multi-insiemi
 1. Concetti di base
 2. Realizzazione con liste
2. Tabelle hash
 1. Funzioni hash
 2. Funzioni hash a partire da tipi di dato qualsiasi: hash-code e funzioni di compressione
 3. Collisioni e loro gestione. Schemi di gestione delle collisioni
 4. Fattore di carico e rehashing

Riferimenti: libro di testo, Sezioni 10.1, 10.2

Mappe ordinate

1. Limiti delle mappe non ordinate: ordinamento rispetto alle chiavi, range query, implementazione efficiente di operazioni su insiemi
2. Operazioni tipiche su mappe ordinate
3. Realizzazione di tabelle ordinate mediante array: realizzazione dell'operazioni tipiche e loro efficienza. Limiti delle tabelle ordinate
4. Rappresentazione di insiemi mediante chiavi ordinate. Realizzazione efficiente di operazioni su insiemi attraverso primitive di tipo merge

Riferimenti: libro di testo, Sezioni 10.3 e 10.5

Alberi binari di ricerca

1. Alberi binari di ricerca: definizione e proprietà generali
2. Realizzazione di mappe ordinate mediante alberi binari di ricerca. Implementazione delle operazioni tipiche di una mappa ordinata. Complessità delle operazioni tipiche e sua dipendenza dall'altezza dell'albero
3. Implementazione di alberi binari di ricerca
4. Alberi binari bilanciati. Definizione di bilanciamento, fattore di bilanciamento
5. Alberi AVL. Definizione e proprietà di bilanciamento. Relazione tra numero di nodi e altezza in un AVL. Limite superiore all'altezza mediante alberi di Fibonacci

Riferimenti:

- Libro di testo, Sezioni 11.1, 11.2 e 11.3
- Appunti del docente pubblicati sul sito del corso

Algoritmi efficienti per il mantenimento di partizioni su insiemi: Union-Find

1. Il problema del mantenimento efficiente di partizioni definite su un insieme comune
2. Implementazione con sequenze e analisi della sua efficienza

3. Implementazione con strutture ad albero. Ottimizzazioni. Efficienza della rappresentazione ottimizzata con struttura ad albero (no prova)

Riferimenti: libro di testo, Sezione 14.7.3

Grafi

1. Grafi: concetti di base e applicazioni. Proprietà elementari di grafi e terminologia. Grafi diretti e non diretti.
2. La struttura dati astratta Grafo: definizione, operazioni consentite. Rappresentazione di grafi e relative strutture dati. Memorizzazione con liste di archi, liste di adiacenza, matrici.
3. Visite di grafi e loro proprietà
 1. Connessione e componenti connesse. Alberi ricoprenti e foreste ricoprenti.
 2. Visita in profondità (DFS)
 3. Realizzazione della visita in profondità in grafi diretti e non diretti
 4. individuazione delle componenti fortemente connesse in grafi diretti e non diretti. Realizzazione in grafi diretti usando la DFS
 5. Chiusura transitiva di grafi. Interpretazione in termini di potenze della matrice di adiacenza. Realizzazione con programmazione dinamica e algoritmo di Floyd-Warshall
 6. Ordinamento topologico di grafi diretti. Ordinamento topologico e grafi diretti aciclici (DAG). Algoritmo di base e sua correttezza. Ordinamento topologico mediante DFS.
4. Cammini minimi
 1. Problemi di percorsi minimi: Single Pair Shortest Path, Single Source Shortest Paths e All Pairs Shortest Paths.
 2. BFS e albero dei cammini minimi da una sorgente in grafi diretti o non diretti non pesati
 3. Proprietà dei cammini minimi e condizioni di ottimalità
 4. Algoritmo di Dijkstra e sua analisi. Fallimento nel caso di presenza di archi con peso negativo
 5. Algoritmo di Bellman-Ford in grafi non diretti con pesi positivi e in grafi diretti. Analisi della complessità computazionale dell'algoritmo di Bellman-Ford. Modifica per rilevare la presenza di cicli con peso negativo
 6. Mantenimento dell'albero dei cammini minimi mediante tabella dei predecessori e conseguente modifica dell'operazione di rilassamento di un nodo

Riferimenti: libro di testo, capitolo 14 da 14.1 a 14.7