Progettazione degli Algoritmi Canale 1 (Lauria)

Programma di esame dettagliato

Massimo Lauria

24/5/2022

Indichiamo il programma del corso con tutto il materiale didattico incluso. Nella lista seguente lo stesso elemento potrebbe essere indicato o elencato più volte. Qualunque cosa sia indicata come inclusa nel riassunto ma non sia presente nella lista dettagliata è comunque inclusa nel programma (fatemi sapere se ne manca qualcosa, in ogni caso).

Di tutti gli algoritmi visti a lezione, è necessario essere in grado di dimostrarne la complessità computazionale.

Riassunto: Il programma di esame include tutto il materiale delle dispense e dei documenti allegati alla pagina

`https://twiki.di.uniroma1.it/twiki/view/Algoritmi2/PALGdiario2014_1`

Include oltrettutto:

- **tutti** gli esercizi presentati nelle dispense (risolti o non risolti nelle stesse);
- **tutte** le sezioni del libro di testo indicate nelle dispense.
- **tutti** gli esercizi del libro indicati nelle dispense.
- **tutti** gli esercizi svolti alle esercitazioni.

Libro di testo:
T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein
Introduzione agli algoritmi e strutture dati 3/ed
McGraw Hill 2010

1 Introduzione del corso

Introduzione al pensiero computazionale, e alla definizione di algoritmo.

- Slide: 001_Introduzione.pdf

Il problema della moltiplicazione di due matrice quadrate: algoritmo standard e l'algoritmo di Strassen.

- Slide: 002_Strassen.pdf
- Testo: sezione 4.2
- Esercizio: Pag. 5 di 002_Strassen.pdf

Il problema della soddisfacibilità delle formule CNF

- Slide: 003_SAT.pdf

Ripasso delle notazioni asintotiche, della complessità degli algoritmi e dei problemi. Esempio di limitazione inferiore alla complessità della ricerca in una sequenza arbitraria. Brevi cenni sulla la testi di Church-Turing (standard ed estesa) e sul modello di calcolo che usiamo per valutare la complessità degli algoritmi.

- Slide: 004_Complessit_dei_Problemi.jpg
- Testo: Capitolo 3

2 Algoritmi di base su grafi

Definizioni dalla teoria dei grafi. Rappresentazione di grafi come liste di archi, matrici di adiacenza e liste di adiacenza. Visita in ampiezza (BFS) e in profondità di grafi semplice e diretti. Teorema delle parentesi e Teorema del cammino bianco. Classificazione di archi in base all'esito della DFS.

Verifica dell'esistenza di cicli in un grafo orientato, e ordinamento topologico di grafi diretti aciclici (DAG). Algoritmo di calcolo dell'ordinamento topologico tramite: estrazione successiva delle sorgenti, e tramite la DFS.

Calcolo delle componenti fortemente connesse utilizzando un algoritmo basato su DFS.

- Slide: 005_BFS.pdf, 010_DFS.pdf, 013_TSORT.pdf, 016_SCOMP.pdf
- Esercizi: pag 5, pag 7, pag 8 in 005_BFS.pdf

- Esercizi: pag 7 in 016_SCOMP.pdf
- Esercizi: 22.2-4, 22.2-5, 22.3-5, 22.3-7, 22.3-12
- Esercizi: 22.4-2, 22.4-3, 22.4-5, 22.5-1, 22.3
- Testo: Appendici B.4 e B.5
- Testo: Capitoli 22.1, 22.2, 22.3, 22.4, 22.5

3 Algoritmi greedy e alberi di copertura minimi

Algoritmi Greedy: introduzione e caratteristiche. Selezione di Attività: le soluzioni ottime di questo problema hanno una sottostruttura ottima che discutiamo con attenzione.

Nozione di grafo pesato, e di albero di copertura. Problema della ricerca dell'albero di copertura minimo. Teorema del taglio. Algoritmi di Kruskal e Prim. Gestione degli insiemi disgiunti per Kruskal. Gestione delle code di Priorità (con heap binario) per Prim.

Il teorema a pagina 9 di 027_UNIONFIND.pdf è dimostrato del capitolo 21.4 del libro di testo. L'enunciato del teorema deve essere compreso, ma la sua dimostrazione non è parte del programma.

Le code di priorità sono spiegate nel capitolo 6.5 del libro ma potrebbe essere necessario, per comprendere tutto, ripassare la struttura dati Heap nei capitoli 6.1-6.3 che non fanno parte del programma. Fa comunque parte del programma la trattazione degli Heap fatta in 028_PRIM_PQUEUE.pdf

- Slide: 021_GREEDY.pdf, 022_SPTREE.pdf, 023_KRUSKAL.pdf
- Slide: 027_UNIONFIND.pdf, 028_PRIM_PQUEUE.pdf
- Slide: 039_ESERCIZIGREEDY.pdf
- Esercizi: pag 4, pag 6 di 021_GREEDY.pdf
- Esercizi: pag 1-2, di 022_SPTREE.pdf
- Esercizi: pag 3, pag 9, pag 11 di 027_UNIONFIND.pdf
- Esercizi: pag 1, pag 5, pag 7 028_PRIM_PQUEUE.pdf
- Esercizi: pag 1-4 039_ESERCIZIGREEDY.pdf
- Esercizi: 16.1-2, 16.1-3, 23.2-1, 23-1

- Esercizi: 21.4-2, 21.4-4, 21.3-4
- Esercizi: 23.2-2, 23.2-8
- Testo: Capitoli 16.1, 21.1, 21.3, 23, 23.1. 23.2, 6.5

4 Cammini minimi

Problema di trovare i cammini minimi in un grafo diretto, pesato, tra coppie di vertici. Le varie tipologie (sorgente singola o multipla, destinazione singola o multipla).

Grafi senza pesi negativi: algoritmo di Dijkstra, visto anche in relazione con la BFS, e con l'algoritmo di Prim.

Rappresentazione dell'albero dei cammini minimi come array dei predecessori.

Grafi con pesi negativi: algoritmo per grafi diretti aciclici e algoritmo di Bellman-Ford per la gestione di cicli negativi.

Proprietà generali degli algoritmi basati sul rilassamento di archi.

Algoritmo di Floyd-Warshall che calcola i cammini minimi tra tutte le coppie di vertici di un grafo, anche in presenza di pesi negativi.

Degli algoritmi B-F e F-W sono incluse nel programma le versioni, lasciate per esercizio, che invece di segnalare semplicemente la presenza di un ciclo negativo, calcolano comunque le distanze per quei vertici non coinvolti nei cicli negativi.

- Slide: 033_CAMMINIMINIMI.pdf 034_CAMMININEGATIVI.pdf
- Slide: 042_FLOYDWARSHALL.pdf
- Esercizi: pag 2, pag 10 di 033_CAMMINIMINIMI.pdf
- Esercizi: pag 3 di 034_CAMMININEGATIVI.pdf
- Esercizi: pag 5 di 042_FLOYDWARSHALL.pdf
- Esercizi: 24.2-2, 24.2-4, 25.3-3, 25.3-4, 25.3-7
- Testo: Capitoli 24.1, 24.2, 24.3, 24.4, 24.6, 25.1, 25.3

5 Algoritmi di approssimazione

Algoritmi di approssimazione per problemi di ottimizzazione che sono intrattabili. Fattore di approssimazione per problemi di massimizzazione e minimizzazione.

Problema del vertex cover. Algoritmo greedy banale, e dimostrazione che il fattore di approssimazione di questo algoritmo è Omega(log n). Algoritmo con fattore di approssimazione 2.

Problema del set cover. Algoritmo di approssimazione greedy per set cover con fattore di approssimazione O(log n).

Numeri armonici: stima della somma dei reciproci come ln(n).

- Slide: 044_APPROX.pdf, 045_SETCOVER.pdf
- Esercizi: pag 11 di 044_APPROX.pdf
- Esercizi: pag 4 di 045_SETCOVER.pdf
- Esercizi: 35.1-4, 35.3-2,
- Testo: Capitoli 35.1, 35.3

6 Divide et impera

Tecnica Divide et Impera. Suddivisione in sottoproblemi indipendenti.

Sottovettore di lunghezza massima. Vari algoritmi fino ad ottenere complessità O(n) con una tecnica divide et impera.

Ripasso del Master theorem per la soluzione delle ricorrenze. Fa parte del programma l'enunciato e la capacità di applicare il teorema. (pag 8 di 051_DIVIDEETIMPERA.pdf e capitolo 4.5)

Algoritmo per l'esponenziazione efficiente di numeri e matrici (vedere esercizio a pag 10 di 051_DIVIDEETIMPERA.pdf).

Ripasso del quicksort (Capitoli 7.1, 7.2, 7.3 del libro) per ricordare la sua strategia divide et impeta e le sue caratteristiche di complessità (caso medio, caso peggiore, ecc…). Non fa parte del programma lo studio dettagliato dell'algoritmo.

Algoritmo divide et impera per la selezione in tempo lineare. Algoritmo randomizzato e algoritmo deterministico.

- Slide: 051_DIVIDEETIMPERA.pdf, 052_SELECTION.pdf
- Esercizi: pag 10 di 051_DIVIDEETIMPERA.pdf
- Esercizi: 9.3-3, 9.3-1, 9.3-8
- Testo: Capitoli 9. (Ripasso su 4.5, 7.1, 7.2, 7.3)

7 Programmazione dinamica

Problema del taglio del bastone, risolto con un algoritmo ricorsivo, con memoizzazione e con approccio bottom up.

Le quattro fasi della progettazione di algoritmi basati sulla programmazione dinamica. Applicazione delle quattro fasi per risolvere il problema della parentesizzazione ottima. Conteggio delle possibili parentesizzazioni.

Formulazione degli algoritmo Bellman-Ford e di Floyd-Warshall dal punto di vista dello schema della programmazione dinamica.

Programmazione dinamica come divite et impera con sottoproblemi ripetuti: memoizzazione come metodo per evitare di risolvere gli stessi sottoproblemi più volte. Approccio bottom-up invece della memoizzazione, e riduzione della memoria utilizzata. Esempio con la sequenza di Fibonacci.

Programmazione dinamica su problemi per stringhe: algoritmi per Longest Common Subsequence e Minimum Edit Distance (Esercizio 15-5(a)). LCS come caso speciale di MED.

Tutti gli esercizi nel file 057_PD4.pdf

- Slide: 054_PD1.pdf, 055_PD2.pdf, 056_PD3.pdf, 057_PD4.pdf
- Esercizi: pag 10 di 054_PD1.pdf
- Esercizi: pag 9, pag 11, pag 12 di 056_PD3.pdf
- Esercizi: pag 1-5 di 057_PD4.pdf
- Esercizi: 15.1-1, 15.1-2, 15.1-3, 15.2-3, 15.2-4
- Esercizi: 15.4-1, 15.4-3, 15.4-4, 15.2
- Esercizi: 15.5(a), 15.5(b)
- Testo: Capitolo 15.1, 15.2, 15.3, 15.4

8 Backtracking

Tutta questa sezione è trattata tramite esempi di risoluzione di esercizi, e tutti gli esercizi fanno parte del programma.

La generazione esaustiva di stringhe binarie di lunghezza fissa, e con vincoli sul numero di uno. Taglio dei rami dell'albero di ricerca. Generazione di matrici binarie.

Concetto di funzione di taglio.

Analisi del numero di foglie, del numero di nodi interni, e della complessità in tutti i nodi, per calcolare la complessità dell'algoritmo e valutarne l'ottimalità.

Esempi generazione di matrici binarie, con e senza restrizioni.

Backtracking per risolvere problemi intrattabili: un problema di decisione: 3-coloring un problema di ottimizzazione: problema dello zaino.

Funzioni di taglio per il problema dello zaino:

- se viene superata la capacità
- se diventa impossibile migliorare una soluzione già trovata

Esempio di uso di una soluzione iniziale prima del backtracking.

Generazione esaustiva di permutazioni, con variazioni. Ricerca di un ciclo Hamiltoniano in un grafo Problema delle regine. Esercizio aperto: la scrittura di un solutore per il sudoku.

- Slide: 058_BT1.pdf 059_BT2HARD.pdf 060_BT3PERM.pdf
- Esercizi: pag 1-9 di 058_BT1.pdf
- Esercizi: pag 3, di 059_BT2HARD.pdf
- Esercizi: pag 1-13 di 060_BT3PERM.pdf
- Testo: non c'è nulla sul libro di testo

9 Esercitazioni in classe

9.1 Esercitazione del 2022.02.28

- Esercizio 1: Trasformare un grafo in una matrice/lista di adiacenza nel caso pesato/non pesato
- Esercizio 2: Vedi esercizio a slide 8 del file 005_BFS.pdf
- Esercizio 3: 22.1-6 del libro di testo

9.2 Esercitazione del 2022.03.07

- Esercizio 1: 22.3-2 del libro di testo
- Esercizio 2: 22.3-12 del libro di testo
- Esercizio 3: 22.1-5 del libro di testo

9.3 Esercitazione del 2022.03.14

- Esercizio 1: Calcolo del trasposto di un grafo orientato a partire dalla list/matrice di adiacenza
- Esercizio 2: 22.5-1 del libro di testo
- Esercizio 3: 22.4-3 del libro di testo

9.4 Esercitazione del 2022.03.21

- Esercizio 1: 22.4-2 del libro di testo

9.5 Esercitazione del 2022.03.28

- Esercizio 1: 16.1-2 e 16.1-3 del libro di testo
- Esercizio 2: In quanti modi è possibile tagliare un grafo?
- Esercizio 3: 23.1-4 del libro di testo
- Esercizio 4: 22.4-5 del libro di testo

9.6 Esercitazione del 2022.04.04

- Esercizio 1: Esempio di applicazione degli algoritmi per Minimum Spanning Tree nel caso dell'algoritmo di clustering HDB Scan

9.7 Esercitazione del 2022.04.11

- Esercizio 1: Applicazione di DFS nel caso di un labirinto
- Esercizio 2: Applicazione dell'algoritmo di Kruskal nel caso di un terreno da irrigare
- Esercizio 3: 23.2-1 del libro di testo

9.8 Esercitazione del 2022.05.02

- Esercizio 1: Esercizio 4 del file 039_ESERCIZIGREEDY.pdf
- Esercizio 2: Algoritmo greedy per trovare l'MST di un grafo con archi di due colori

9.9 Esercitazione del 2022.05.09

- Esercizio 1: Esercizio 35.1-4 del libro di testo
- Esercizio 2: Determinare il valore dei pesi degli archi di un grafo affinchè sia possibile calcolare il cammino minimo tra due vertici

9.10 Esercitazione del 2022.05.22

- Esercizio 1: Calcolare il numero di inversioni all'interno di un array di interi utilizzando la tecnica del Divide et Impera
- Esercizio 2: Esempio di applicazione del Critical Path Method (modellazione del grafo delle operazioni e cammino minimo su DAG)

9.11 Esercitazione del 2022.05.23

- Esercizio 1: Stampare tutte le stringhe binarie di lunghezza n, con un numero di 1 compreso tra k_1 e k_2.
- Esercizio 2: Esercizio di programmazione dinamica a pagina 5 del file 057_PD4.pdf
- Esercizio 3: Esercizio di programmazione dinamica a pagina 4 del file 057_PD4.pdf
- Esercizio 4: Stampare tutte le sequenze di numeri strettamente crescenti e lunghe n, contenenti numeri da 0 a $n + k - 1$ per $n > 0$ e $k > 0$