Corso di Laurea in Fisica
a.a 2005-2006
INFORMATICA II
Titolare: Prof.Roberto Grossi

Programma.

CONTENUTI DEL CORSO:

Introduzione al modello di calcolo, all'analisi e alla complessita' degli algoritmi. Algoritmi ricorsivi e relazioni di ricorrenza: divide et impera e programmazione dinamica. Strutture di dati combinatorie con applicazioni: algoritmi per array, liste, alberi, grafi, pile, code, code di priorita', dizionari. Problemi P, NP, e NP-completi.

Programma in dettaglio:

1 Problemi computazionali
- Indecibilita' di problemi computazionali
- Trattabilita' di problemi computazionali (rappresentazione e dimensione dei dati, algoritmi polinomiali ed esponenziali)
- Problemi NP-completi
- Modello RAM e complessita' computazionale

2 Sequenze: array
- Sequenze lineari: modalita' di accesso e allocazione della memoria
- Opus: scheduling della CPU (ordinamento per selezione e per inserimento)
- Complessita' di problemi computazionali (limiti superiori e inferiori)
- Ricerca di una chiave (ricerca binaria)
- Ricorsione e paradigma del divide et impera (moltiplicazione veloce di numeri, ordinamento per fusione, ordinamento e selezione per distribuzione)
- Alternativa al teorema fondamentale delle ricorrenze
- Array di dimensione variabile
- Opus: array a piu' dimensioni e matrici nella grafica (moltiplicazione veloce, sequenza ottima di moltiplicazioni)
- Paradigma della programmazione dinamica (sottosequenza comune piu' lunga, partizione di un insieme di interi, problema della bisaccia, pseudo-polinomialita')

3 Sequenze: liste
- Liste (ricerca, inserimento e cancellazione, liste doppie e liste circolari)
- Opus: problema dei matrimoni stabili
- Unione e appartenenza a liste disgiunte
- Liste ad auto-organizzazione
- Tecniche di analisi ammortizzata
- Liste randomizzate

4 Alberi
- Alberi binari (algoritmi ricorsivi, inserimento e cancellazione)
- Opus: minimo antenato comune
- Visita per ampiezza: rappresentazione implicita e succinta (rank, select, limite inferiore sullo spazio)
- Alberi Cardinali, Alberi Ordinali e Parentesi Bilanciate

5 Grafi
- Grafi (alcuni problemi, rappresentazione, cammini minimi e chiusura transitiva mediante moltiplicazione di matrici)
- Opus: colorazione di grafi (assegnazione delle lunghezze d'onda e grafi a intervalli)
- Grafi generati casualmente e modelli di reti complesse (Erdos-Renyi, effetto di piccolo mondo,

invarianti di scala,
instradamento su reti a piccolo mondo)
- Opus: motori di ricerca e classificazione dei documenti (navigatore casuale e page rank, catene di Markov, calcolo del page rank)

6 Pile e code
- Pile (implementazione mediante array e mediante riferimenti)
- Opus: espressioni in notazione polacca
- Code (implementazione mediante array e mediante riferimenti)
- Opus: visite di grafi (ampiezza, profondita')
- Grafi diretti aciclici e ordinamento topologico, componenti (fortemente) connesse

7 Code con priorita'
- Code con priorita' (heap e ordinamento)
- Cammini minimi (Floyd-Warshall, Dijkstra)
- Alberi minimi di ricoprimento (Kruskal, Prim)
- Leftist Heap, code binomiali

8. Dizionari
- Alberi binari di ricerca (AVL)
- B-alberi
- Tabelle hash
- Trie
- Opus: ricerca testuale e ordinamento di suffissi

9. NP-completezza
- Definizione delle classi P, NP, NPC
- Riduzioni polinomiali
- Algoritmi di approssimazione

TESTI DI CONSULTAZIONE:

1. C. Demestrescu, I. Finocchi, G. F. Italiano.
Algoritmi e strutture dati, McGraw Hill, 2004.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein.
Introduction to algorithms, MIT Press, second edition, 2002
3. Dispense consegnate a lezione.

OBIETTIVI FORMATIVI:

Definire formalmente la nozione di algoritmo e di modello di calcolo.
Caratterizzare i dati da elaborare, organizzandoli e strutturandoli nel modo piu' opportuno al fine di agevolarne l'uso da parte degli algoritmi. Progettare algoritmi corretti (che risolvono cioe' sempre e solo il problema a cui si e' interessati) ed efficienti (cioe' che lo risolvono il piu' velocemente possibile o usano il minor spazio di memoria possibile), attraverso l'esame di diversi paradigmi. Studiare le limitazioni inerenti dei problemi da risolvere, in particolare di quelli la cui soluzione richiede l'esame di tutte le possibilita'.

PREREQUISITI:

Conoscenza di un linguaggio di programmazione (C, C++, Java)

METODI DIDATTICI:

Lezioni frontali con esercitazioni.
Sviluppo di codice in laboratorio.
Uso di strumenti di visualizzazione.

MODALITA' DI VERIFICA DELL'APPRENDIMENTO:

Esercizi scritti di esame, orali di verifica,
sviluppo di progetti.

ALTRE INFORMAZIONI:

sito web: <http://www.di.unipi.it/~grossi/ALGM>