Corso di laurea in Fisica
INFORMATICA II_ (5 crediti, II semestre)
A.A. 2007-2008
Titolare:Prof. Roberto Grossi)http://www.di.unipi.it/~grossi/ALGM

_Dipartimento di Informatica__ **INIZIO LEZIONI E ORARIO**

- Inizio delle lezioni: secondo semestre del corso di Laurea in Matematica
- Orario: da definire.
- Il corso è mutuato da "Algoritmi e strutture di dati" a Matematica ed è possibile presentarsi all'esame con un programma ridotto da concordare insieme. _

MOTIVAZIONI

"Fino a poco tempo fa, i matematici teorici consideravano un problema risolto se esisteva un metodo conosciuto, o algoritmo, per risolverlo; il procedimento di esecuzione dell'algoritmo era di importanza secondaria._ Tuttavia, c'è una grande differenza tra il sapere che è possibile fare qualcosa e il farlo. Questo atteggiamento di indifferenza sta cambiando rapidamente, grazie ai progressi della tecnologia del computer. Adesso, è importantissimo trovare metodi di soluzione che siano pratici per il calcolo._ La teoria della complessità studia i vari algoritmi e la loro relativa effficienza computazionale. Si tratta di una teoria giovane e in pieno sviluppo, che sta motivando nuove direzioni nella matematica e nello stesso tempo trova applicazioni concrete quali quello fondamentale della sicurezza e identificazione dei dati."_ -- E. Bombieri, Medaglia Fields, in *La matematica nella società di oggi*, 2001

OBIETTIVI FORMATIVI

Definire formalmente le nozioni di algoritmo e di modello di calcolo caratterizzandone gli aspetti rilevanti. Organizzare e strutturare i dati da elaborare nel modo più opportuno al fine di agevolarne l'uso da parte degli algoritmi. Progettare algoritmi corretti (che risolvono cioè sempre e solo il problema a cui si è interessati) ed efficienti (cioè che lo risolvono il più velocemente possibile o usano il minor spazio di memoria possibile), attraverso l'esame di paradigmi diversi e problemi provenienti dal mondo reale. Studiare le limitazioni inerenti dei problemi da risolvere, in particolare di quelli la cui soluzione richiede l'esame di tutte le possibilità.

PREREQUISITI E METODOLOGIA

- Conoscenza di un linguaggio di programmazione (C, C++, Java).
- Lezioni frontali con esercitazioni.
- Sviluppo di codice in laboratorio.
- Uso di strumenti di visualizzazione.
- Verifica tamite esercizi scritti di esame, orali e sviluppo di progetti.

_CONTENUTI

Introduzione al modello di calcolo, all'analisi e alla complessità 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 priorità e dizionari. Problemi P, NP e NP-completi.

PROGRAMMA DETTAGLIATO (alcuni argomenti sono facoltativi)

1. **Problemi computazionali.** Indecibilità di problemi computazionali - Trattabilità di problemi computazionali (rappresentazione e dimensione dei dati, algoritmi polinomiali ed esponenziali) - Problemi NP-completi - Modello RAM e complessità computazionale.
2. **Sequenze: array.** Sequenze lineari: modalità di accesso e allocazione della memoria - Opus: scheduling della CPU (ordinamento per selezione e per inserimento) - Complessità 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 più dimensioni e matrici nella grafica (moltiplicazione veloce, sequenza ottima di moltiplicazioni) - Paradigma della programmazione dinamica (sottosequenza comune più lunga, partizione di un insieme di interi, problema della bisaccia, pseudo-polinomialità).

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, profondità) - Grafi diretti aciclici e ordinamento topologico, componenti (fortemente) connesse.
7. **Code con priorità.** Code con priorità (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 E MATERIALE DIDATTICO

- P. Crescenzi, G. Gambosi, R. Grossi, *Strutture di Dati e Algoritmi*, Addison-Wesley Pearson, 2006.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. *Introduction to Algorithms*, MIT Press, second edition, 2001.