Algoritmi

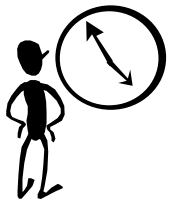
a. a. 2012-2013

Docenti corso

docente	matricola
Ugo Vaccaro	= 0 mod 3
Alfredo De Santis	= 1 mod 3
Marcella Anselmo	= 2 mod 3

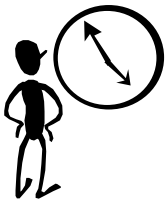
Il modulo 3 deve essere effettuato dividendo la matricola senza prefisso

Orari Corso



- Martedì 16:00 - 18:00, aula F/4
- Venerdì 11:00 - 13:00, aula F/4

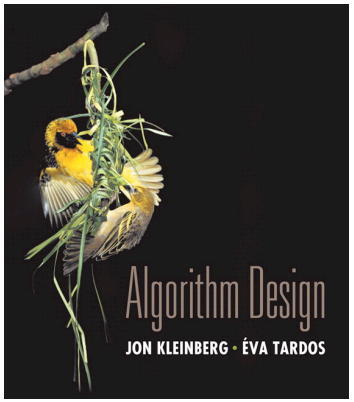
Orari Ricevimento Studenti



- Martedì 15:00 - 16:00
- Venerdì 15:00 - 17:00
- Previo accordo

Libro di testo

Jon Kleinberg, Éva Tardos,
Algorithm Design
Addison-Wesley, 2005

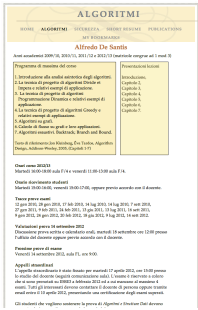


Slide

Kevin Wayne, Princeton University
http://www.cs.princeton.edu/~wayne/kleinberg-tardos/

http://www.dia.unisa.it/professori/ads/

- Ci sono le slide dell'a.a. 2011/2012
- Saranno aggiornate di volta in volta


ALGORITMI

Alfredo De Santis

Modalità esame

- Prova scritta
- Prova orale

Modalità esame

- Prova scritta
- Prova orale

- Calendario prova scritta 2013
  - Preappello: dal 7 al 16 Gennaio (solo per i corsi tenuti nel I semestre 2012/13)
  - I appello; dal 17 Gennaio al 6 Febbraio (per tutti i corsi)
  - II appello: dal 7 al 27 Febbraio (per tutti i corsi)
  - Giugno
  - Luglio
  - Settembre
  - Novembre (se mancano al massimo 4 esami)

Obiettivi formativi del corso

- fornire allo studente metodi e conoscenze atte al progetto di algoritmi efficienti;
- fornire strumenti per l'analisi delle risorse (spazio e tempo) utilizzate da algoritmi;
- fornire un catalogo dei più noti ed efficienti algoritmi per problemi computazionali di base (ordinamento, ricerca, ottimizzazione di risorse, etc.)

- Critical thinking e problem-solving.

Approccio ai problemi

- Descrizione del problema computazionale reale
- Modellizzazione del problema reale mediante un problema astratto
- Risoluzione del problema astratto mediante un algoritmo ottenuto attraverso l'applicazione delle tecniche generali di progetto di algoritmi introdotte nel corso
- Analisi delle risorse utilizzate dall'algoritmo elaborato

Algoritmo

- [webster.com] A procedure for solving a mathematical problem (as of finding the greatest common divisor) in a finite number of steps that frequently involves repetition of an operation.

- [Knuth] An algorithm is a finite, definite, effective procedure, with some input and some output.

Etimologia

Abu'Abd Allah Muhammad ibn Musa al-Khwarizm, matematico persiano del IX secolo scrisse Kitab al-jabr wa'l-muqabala

- "Libro sulla ricomposizione e sulla riduzione"
- Soluzioni equazioni lineari e quadratiche
- dal quale tra l'altro prende anche le origini la parola algebra

Libro su sistemi posizionamento decimale,

- tradotto in latino nel XII secolo Algoritmi de numero Indorum

traduzione in latino di al-Khwarizm

Applicazioni

Ampio spettro di applicazioni:

- Caching.
- Compilers.
- Databases.
- Scheduling.
- Networking.
- Data analysis.
- Signal processing.
- Computer graphics.
- Scientific computing.
- Operations research.
- Artificial intelligence.
- Computational biology.
- . . .

Il corso è focalizzato su tecniche algoritmiche che sono utili in pratica

Programma: capitoli libro

1. Introduction: Some Representative Problems
2. Basics of Algorithms Analysis
3. Graphs
4. Greedy Algorithms
5. Divide and Conquer
6. Dynamic Programming
7. Network Flow
8. NP and Computational Intractability
9. PSPACE: A Class of Problems Beyond NP
10. Extending the Limits of Tractability
11. Approximation Algorithms
12. Local Search
13. Randomized Algorithms

Programma: capitoli libro

1. Introduction: Some Representative Problems
2. Basics of Algorithms Analysis
3. Graphs
4. Greedy Algorithms
5. Divide and Conquer
6. Dynamic Programming
7. Network Flow
8. NP and Computational Intractability
9. PSPACE: A Class of Problems Beyond NP
10. Extending the Limits of Tractability
11. Approximation Algorithms
12. Local Search
13. Randomized Algorithms

Differenze corsi anni scorsi

- Fino al 2008/09:
  - Algoritmi e Strutture Dati
  - Laboratorio di Algoritmi e Strutture Dati
- Dal 2009/10:
  - Algoritmi
  - Strutture Dati

- {Algoritmi} – {Algoritmi e Strutture Dati}
  - Algoritmi su grafi
  - Algoritmi esaustivi
- {Algoritmi e Strutture Dati} - {Algoritmi}
  - Strutture Dati (Code a priorità, Alberi ricerca, Alberi rosso-neri, Insiemi disgiunti, Tabelle Hash)
  - Complessità Computazionale

Algoritmo di Euclide

Descritto negli *Elementi* di Euclide (circa 300 A. C.)

Serve a calcolare il Massimo Comun Divisore

gcd(30,21) = ?

gcd(63,30) = ?

gcd(4864,3458) = ?

L'edizione 1570

Algoritmo di Euclide

Descritto negli *Elementi* di Euclide (circa 300 A. C.)

Serve a calcolare il Massimo Comun Divisore

gcd(30,21) = 3

gcd(63,30) = 3

gcd(4864,3458) = 38

L'edizione 1570

Prima idea

- Prima fattorizzare
- Fattori comuni con esponente più piccolo

30 = 2x3x5
21 = 3x7
gcd(30,21) = 3

30 = ?
63 = ?
gcd(63,30) = ?

Prima idea

- Prima fattorizzare
- Fattori comuni con esponente più piccolo

30 = 2x3x5
21 = 3x7
gcd(30,21) = 3

30 = 2x3x5
63 = 3^2x7
gcd(63,30) = 3

Fattorizzazione

Dato n calcolare l'unica fattorizzazione

$$n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$$

con p_i primo ed $e_i \geq 0$

Fattorizzazione

Dato n calcolare l'unica fattorizzazione

$$n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$$

con p_i primo ed $e_i \geq 0$

Quanto è difficile?

Difficoltà Fattorizzazione

- 200 cifre decimali
- Fattorizzato 9 maggio 2005 da F. Bahr, M. Boehm, J. Franke, e T. Kleinjung
- Tempo equivalente al lavoro di 75 anni di un singolo computer con processore 2.2 GHz AMD Opteron e 2 GB RAM

RSA-200 =

27997833911221327870829467638722601621070446786955428537560009929326128400107609345671052955360856061822351910951365788637105954482006576775098580557613579098734950144178863178946295187237869221823983

RSA-200 =

35324619344027701212726049781984643686711974001976250236493034687761212536794232000585479565280883491

X

79258699544783330333470858414800596877379758573642199607343303414557678728181521353814093047401854671

Difficoltà Fattorizzazione

- 232 cifre decimali, ovvero 768 bit
- Fattorizzato 12 dicembre 2009 da Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thomé, Joppe W. Bos, Pierrick Gaudry, Alexander Kruppa, Peter L. Montgomery, Dag Arne Osvik, Herman te Riele, Andrey Timofeev, e Paul Zimmermann
- Tempo equivalente al lavoro di 2.000 anni di un singolo computer con processore 2.2 GHz AMD Opteron e 2 GB RAM

RSA-768 =

12301866845301177551304949583849627207728535695953347921973224521517264005072636575187452021997864693899564749427740638459251925573263034537315482685079170261221429134616704292143116022212404792747377940806653514195974598569021434 13

RSA-768 =

33478071698956898786044169848212690817704794983713768568912431388982883793878002287614711652531743087737814467999489

X

36746043666799590428244633799627952632279158164343087642676032283815739666511279233373417143396810270092798736308917

Difficoltà Fattorizzazione

Ancora da fattorizzare:

RSA-896

RSA-1024

RSA-1536

RSA-2048

Algoritmo di Euclide

Descritto negli *Elementi* di Euclide (circa 300 A. C.)

Teorema della ricorsione del gcd
Per tutti gli interi $a \geq 0$ e $b > 0$
$\gcd(a,b) = \gcd(b, a \bmod b)$

Algoritmo di Euclide

Descritto negli *Elementi* di Euclide (circa 300 A. C.)

Teorema della ricorsione del gcd
Per tutti gli interi $a \geq 0$ e $b > 0$
$\gcd(a,b) = \gcd(b, a \bmod b)$

```
Euclide (a,b)
if   b = 0  then return a
          else  return   Euclide (b, a mod b)
```


Algoritmo di Euclide: Esempi

Euclide (30,21) = Euclide (21,9)
= Euclide (9,3)
= Euclide (3,0) = 3

Euclide (63,30) = Euclide (30,3)
= Euclide (3,0) = 3

Algoritmo di Euclide: Esempi

Euclide (4864,3458) = Euclide (3458,1406)
= Euclide (1406,646)
= Euclide (646,114)
= Euclide (114,76)
= Euclide (76,38)
= Euclide (38,0) = 38

Algoritmo di Euclide

- Correttezza
- Termina sempre?
- Efficienza: Ma quanto tempo impiega?

Algoritmo di Euclide: complessità

- Assumiamo $a \geq b$
- Al massimo $\log b$ chiamate
- Per ogni chiamata $O((\log a)^2)$ operazioni su bit
- Totale: al massimo $O((\log a)^3)$ operazioni su bit
- Euclide (a,b) richiede al massimo $O((\log a)^2)$ operazioni su bit