Quello che i calcolatori possono e non possono fare

Le teorie della calcolabilità e della complessità

Prof. Ugo de'Liguoro

unito.it
DIPARTIMENTO DI INFORMATICA
Corsi di Studi in informatica
C.so Svizzera 185, Torino

unito.it
DIPARTIMENTO DI INFORMATICA
Corsi di Studi in informatica
C.so Svizzera 185, Torino

"Se mettete il software appropriato in un computer, questo farà tutto quello che desiderate. Le macchine in sé possono avere dei limiti, ma non ci sono limiti a quello che i software possono realizzare".

Sbagliato, clamorosamente sbagliato. Tanto sbagliato da farmi dire che il libro che avete per le mani può essere riassunto in poche parole come il tentativo di confutare, anzi di demolire questa affermazione.

D. Harel, Computer a responsabilità limitata, 2000.

Domande e risposte

Un problema algoritmico è definito da:

- un insieme di domande o input ammissibili
- un insieme di risposte o output
- una relazione che collega ogni domanda ad una o più risposte corrette

Problemi matematici

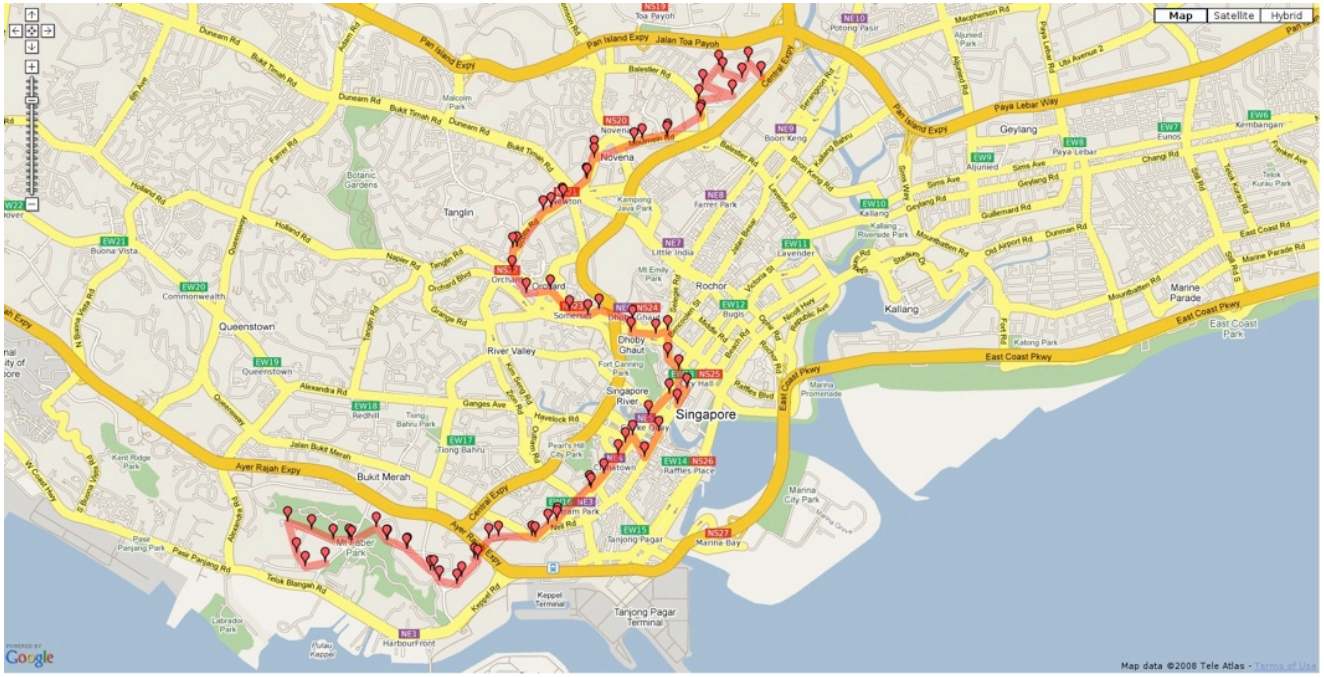
✦ Fattorizzazione: dato n ∈ Nat trovare la sua scomposizione in fattori primi

✦ Equazioni diofantine (X Prob. Hilbert): data un'equazione a coefficienti interi in un certo numero di variabili, stabilire se esiste una soluzione ossia una sostituzione delle variabili con interi che la renda vera:

$1027x + 712y = 1$ ha soluzioni $x = -165$ ed $y = 238$

$x^2 - 3(y+1)^2 = 1$ non ha soluzioni

Il cammino minimo (MINPATH)

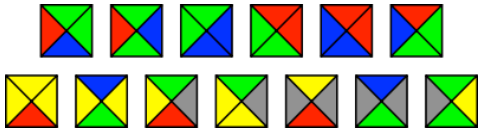


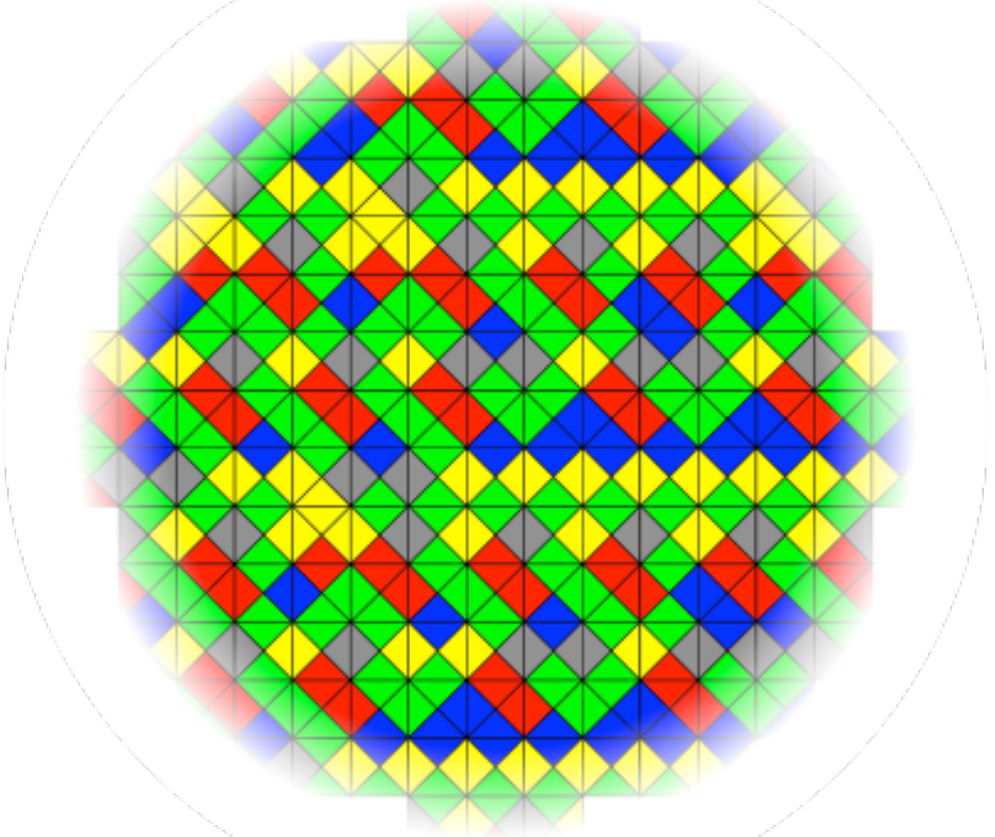
Dati due punti su di una cartina stradale che siano raggiungibili l'uno dall'altro trovare il percorso più breve dal primo al secondo punto

Il commesso viaggiatore (TSP)

Data una carta stradale ed n città tutte connesse tra loro, trovare il percorso più breve tale che ciascuna città sia raggiunta esattamente una volta.

Problema delle piastrelle (TILE)





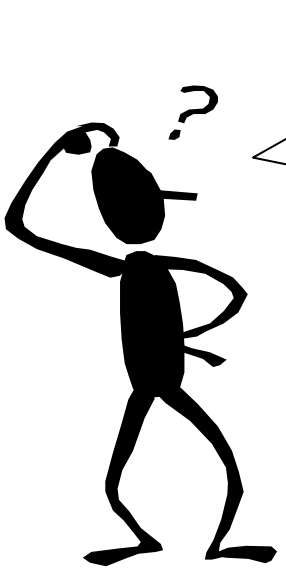
Dato un numero finito di fogge di piastrelle quadrate, decidere se è possibile piastrellare qualunque stanza disponendo le piastrelle in modo che i lati combacianti abbiano colori uguali.

Questi non sono "problemi"

- Come sbarcare il lunario?
- Che cos'è il sommo bene?
- Quanto fa 7/0?

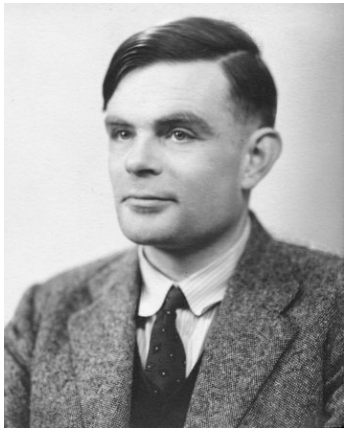
Quando un problema è risolubile?

Un problema algoritmico è risolubile quando esiste un algoritmo che per ogni domanda calcola una risposta corretta.



Che cosa è un algoritmo?

Un procedimento meccanico capace di rispondere in un numero finito di passi ad ogni domanda del problema



Alan M. Turing

Prime conseguenze

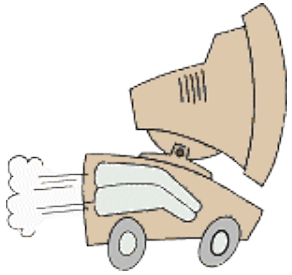
Ci sono più funzioni da Nat in Nat che algoritmi:

- infatti ogni algoritmo deve essere esprimibile attraverso un modello di calcolo, ad esempio un linguaggio di programmazione
- le "frasi" di un tale linguaggio, ossia i programmi, sono stringhe finite formate da simboli tratti da un vocabolario finito, quindi sono numerabili
- le funzioni da Nat a Nat sono più che numerabili!

Indecidibilità della fermata (HALT)

Probema della fermata:

- input: un programma P ed un suo ingresso x
- output:
  - SI quando P(x) termina la sua esecuzione
  - No quando l'esecuzione di P(x) non si ferma mai



Altri problemi indecidibili

❖ il problema TILE è indecidibile, ma semidecidibile (esiste un algoritmo che si ferma con successo esattamente tutte le volte che una domanda ha risposta SI)

❖ il X problema di Hilbert (equazioni diofantine) è indecidibile ma semidecidibile

❖ l'eguaglianza di due numeri reali è indecidbile anche se i reali sono "ricorsivi", ossia se esiste un algoritmo per generarne lo sviluppo decimale con precisione arbitraria.

Problemi ancora più difficili

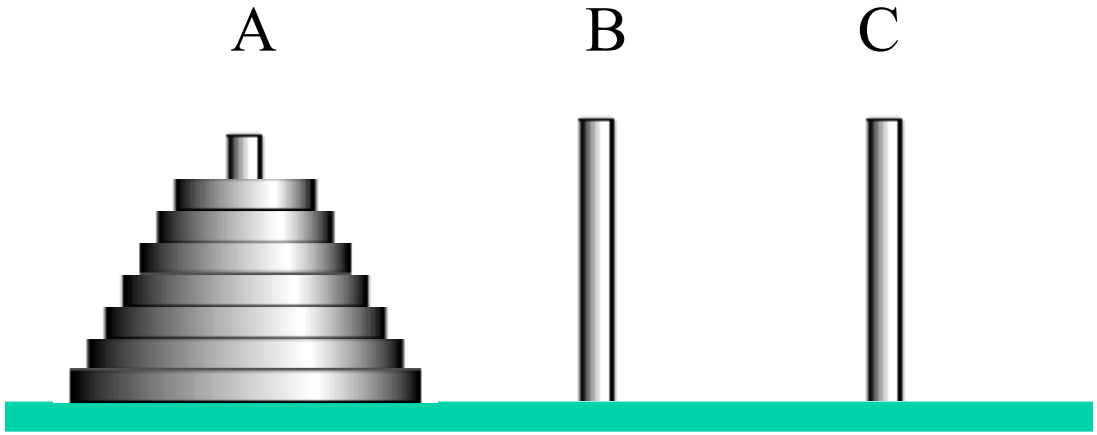
- **Totalità**: dato un programma P è vero che P(x) termina per ogni x?
- **Equivalenza**: dati i programmi P e Q è possibile stabilire se sono equivalenti?

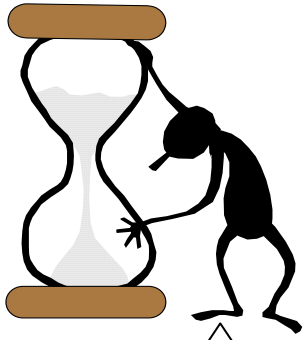
Ma sono insolubili!

Questi problemi resterebbero insolubili anche se ci fosse un mago capace di decidere il problema HALT

Calcolabile in teoria ed in pratica

Dati tre pioli, su cui sono inseriti n dischi di diametro crescente, spostare la torre da un piolo sorgente (A) ad uno destinazione (B) , sfruttando un piolo d'appoggio (C), muovendo un disco alla volta, senza mai sovrapporre un disco più grande ad uno più piccolo.





Ci vogliono 2^n-1 mosse per spostare una torre di n

La classe PTIME (brevemente P)

Le funzioni esponenziali, come 2^n, crescono molto più rapidamente di quelle polinomiali, come n^k:

> PTIME = la classe dei problemi algoritmici risolubili in tempo n^k per qualche k.

Tutti i problemi che sappiamo risolvere in modo efficiente sono risolubili da algoritmi che lavorano in tempo polinomiale, ossia sono in P.

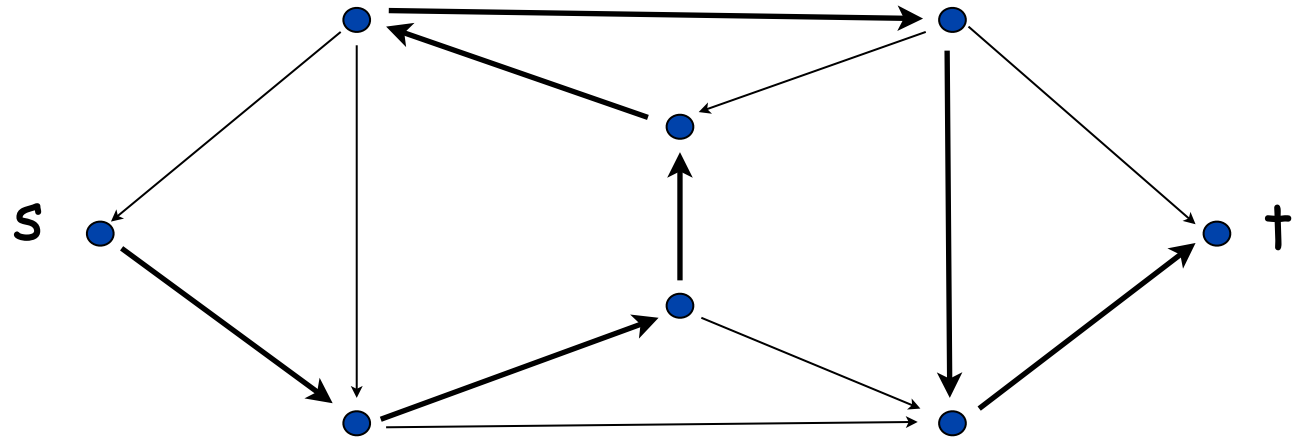
PATH è in P

INPUT: un grafo G, due vertici s e t

1. marca s
2. finché ci sono nodi non marcati adiacenti a qualche nodo marcato ripeti 3
3. marca un tale nodo
4. rispondi SI se t è marcato, NO altrimenti.

Ricerca e verifica

Consideriamo il problema di riconoscere se in un grafo G e dati due vertici s e t esiste un cammino da s a t che include tutti i vertici esattamente una volta (cammino hamiltoniano):

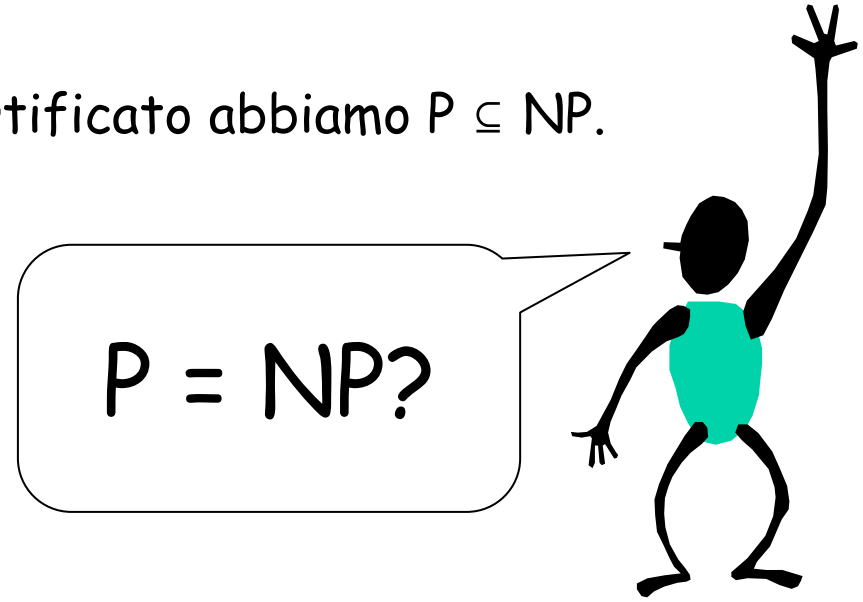


Dato un cammino la verifica che sia hamiltoniono è PTIME, ma non conosciamo un algoritmo che sappia trovarlo in tempo polinomiale.

La classe NPTIME (brev. NP)

NPTIME = la classe dei problemi algoritmici le cui risposte (certificati) sono verificabili in tempo n^k per qualche k.

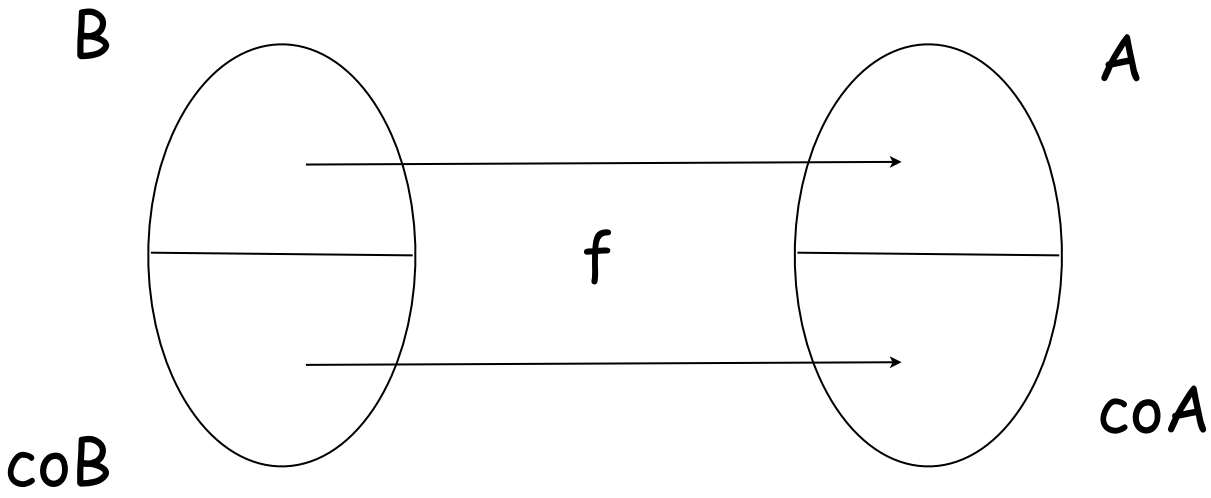
Visto che una computazione è un certificato abbiamo $P \subseteq NP$.



Problemi NP-completi

Un problema A è NP-completo se

1. A è in NP
2. ogni B $\in$ NP si riduce in tempo polinomiale a A



Problemi NP-completi

- I problemi NP-completi sono risolubili in tempo esponenziale
- Non sappiamo se siano risolubili in tempo polinomiale, ma se uno solo di essi lo fosse (cioè appartenesse a P) allora P = NP.

I problemi NP-completi sono numerosi:

- la soddisfacibilità di una formula della logica proposizionale
- l'esistenza di un cammino hamiltoniano
- problemi legati a giochi come il SUDOKU

Il rovescio della medaglia

Il fatto che ci siano cose che non sappiamo fare coi computer in modo efficiente può essere utile:

I sistemi criptografici a chiave pubblica si basano sulla difficoltà di fattorizzare interi con molte cifre ....