

TECNICHE DI ANALISI

BISOGNA DISTINGUERE TRA PROBLEMA ALGORITMICO E ALGORITMO.

IL PROBLEMA ALGORITMICO E' QUELLO CHE VOGLIAMO RISOLVERE, COME AD ESEMPIO:

- TROVARE IL MASSIMO DI UNA SEQUENZA
- TROVARE IL CAMMINO MINIMO DI UN GRAPPO
- ORDINARE UNA SEQUENZA

L'ALGORITMO E' QUELLO CHE RISOLVE IL PROBLEMA.

PROBLEMA ALGORITMICO

ABBIAMO LA SEGUENTE RELAZIONE: $P \rightarrow I \times S$ DOVE:

- I INSIEME DEGLI INPUT, O INSERIMENTI DEL PROBLEMA
- S INSIEME DELLE SOLUZIONI

PER OGNI INSIEMA I DELL'INPUT HA UNA SOLUZIONE O UN'INSERIMENTO DELLE POSSIBILI SOLUZIONI S CHE SI VOLGONO OTTENERE.

ALGORITMO

DESCRIZIONE: PRECISA E NON AMBOGA DI UN PROCEDIMENTO DI CALCOLO CHE PUO' ESSERE ESEGUITO "MECCANICAMENTE" SIA AGLIUMO SIA DALLA MACHINA.

COSTRUZIONE DI UN ALGORITMO

BISOGNA PRENDERE IN CONSIDERAZIONE VARI FATTORI PER COSTRUZIONE DEL COSTRUTTO DI UN ALGORITMO RISPETTO AD UN PROBLEMA CONSIDERANDO L'ALGORITMO $A : I \rightarrow S$. E' UNA FUNZIONE CHE HA:

- I : INSIEME DEGLI INPUT
- S : INSIEME DELLE SOLUZIONI

PER OGNI INPUT i , $A(i)$ E' UNA SOLUZIONE DI P . QUESTA E' UNA FUNZIONE PARZIALE: APPARTENE QUNDO A UN INSIEME DI FUNZIONI CHE POSSONO NON ESSERE DEFINITE PER CERTI ARGOMENTI, AD ESEMPIO:

- OPERAZIONI PARZIALI SU STRUTTURE DATI CHE PROVOCANO ERRORE DINAMICO [POP() SU STACK VUOTO]
- Algoritmi CHE NON TERMINANO (LOOP INFINITO)

ABBIAMO DUE TECNICHE FONDAMENTALI NEGLI ALGORITMI:

- 1) INIZIATIVA: PER ALGORITMI RICORSIVI
- 2) INVARIANTI DI CICLO: PER ALGORITMI ITERATIVI

EFFICIENZA DI UN ALGORITMO

UNO STESSO PROBLEMA PUO' ESSERE RISOLTO DA UN ALGORITMO, CHE POSSONO RICORDARE UN NUMERO DI OPERAZIONI DIFFERENTI E UNO STESO AI MENOVA DIFFERENTI TRA LORO.

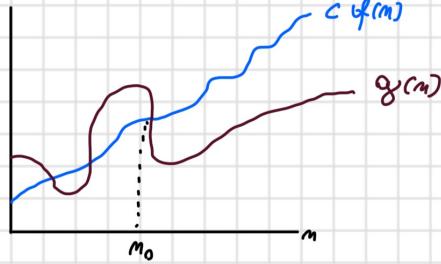
BIG O: CLASSE DI ALGORITMI PER I CERIMENTI IN TEMPO DI:

- SPAZIO DI MEMORIA
 - TEMPO DI CALCOLO
- } NECESSARI PER PRODURRE UN'OUTPUT

ANDAMENTO ASINTOTICO: BIG O, THETA E OMEGA

BIG O

UNA FUNZIONE $g(m)$ APPARTENE ALL'INSIEME $O(f(m))$ SE ESISTONO DUE COSTANTI $C > 0$ E $m_0 \geq 0$ TALI CHE $g(m) \leq Cf(m)$ PER OGNI $m \geq m_0$, OSSIA, DA UN CERTO PUNTO IN AVANTI, g È UNA FUNZIONE "MULTPLA" DI f (CREDE AL PIÙ COME f)

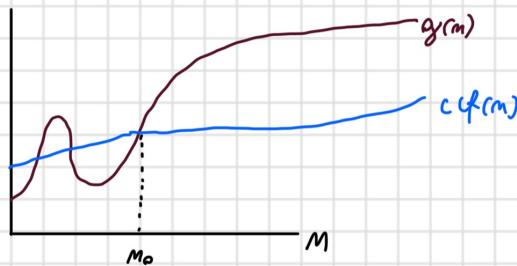


$$g(m) = O(f(m))$$

↓
AGURO DI DEMONSTRARE, NON RAPPRESENTA UGLIAGGIANZA

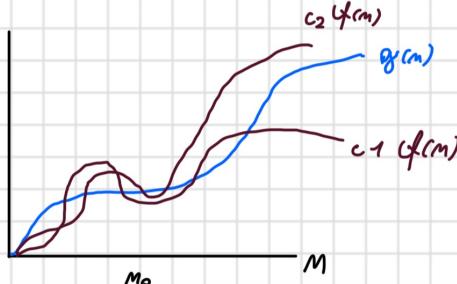
OMEGA

UNA FUNZIONE $g(m)$ APPARTENE ALL'INSIEME $\Omega(f(m))$ SE ESISTONO DUE COSTANTI $C > 0$ E $m_0 \geq 0$ TALI CHE $g(m) \geq Cf(m)$ PER OGNI $m \geq m_0$, OSSIA, DA UN CERTO PUNTO IN AVANTI, g È UNA FUNZIONE "SOMMULPA" DI f (CREDE AL PIÙ COME f)



THETA

UNA FUNZIONE $g(m)$ APPARTENE ALL'INSIEME $\Theta(f(m))$ SE ESISTONO TRE COSTANTI $c_1, c_2 > 0$ E $m_0 \geq 0$ TALI CHE $c_1 f(m) \leq g(m) \leq c_2 f(m)$ PER OGNI $m \geq m_0$, OSSIA, DA UN CERTO PUNTO IN AVANTI, f È COMPRENSA TRA UNA "MULTPLA" DI f E UNA "SOMMULPA" DI f



PROPRIETÀ DI BIG-O, MIGRA IN OMEGA

- TRASPOSTA: $f(m) \in \Theta(g(m))$

- SIMMETRICA: SOLO PER TUTTO $\Rightarrow f(m) \in \Theta(g(m)) \Leftrightarrow g(m) \in \Theta(f(m))$

- SIMMETRIA TRASPOSTA: PER BIG-O E OMEGA $\Rightarrow f(m) \in O(g(m)) \Leftrightarrow g(m) \in \Omega(f(m))$

- SOMMA: $f(m) + g(m) \in \Theta(\max\{f(m), g(m)\})$ ANCHE LOCALMENTE PER UN $i \in \Theta \Rightarrow$ SE SONO DUE FUNZIONI ASSINTOTICHE $f(m) \leq g(m)$, LA UNA CRESCE E' DOMINATA DA QUESTA PIÙ GRANDE TRA LE DUE. QUANDO M CRESCIE MOLTO, IL TERMINE PIÙ GRANDE "VINCIT" E LA SOMMA SI COMPORTA COME LUI.

Comportamenti notevoli Trattabili

$\Theta(1)$	costante
$\Theta(\log n)$	logaritmico
$\Theta(\log n)^k$	poli-logaritmico
$\Theta(n^{1/2})$	radice quadrata
$\Theta(n)$	lineare
$\Theta(n \log n)$	pseudolineare
$\Theta(n^2)$	quadratico
$\Theta(n^3)$	cubico

Intrattabili

$\Theta(2^n)$	esponenziale
$\Theta(n!)$	fattoriale
$\Theta(n^n)$	esponenziale in base n

Complessità di Algoritmi e Problemi

IN GENERALE SIA IL NUMERO DI PASSI ELEMENTARI CHE LO SPAZIO ALLOCATO DA UN ALGORITMO A DEPENDERE DALLA DIMENSIONE M DELL'INPUT. TUTTAVIA, IN GENERALE NON SONO FUNZIONI DI M, PERCHÉ A PARTE DI DIMENSIONE, INPUT DIVERSE POSSONO RICHIEDERE UN TEMPO DI CALCOLO E UN'OCCUPAZIONE DI MEMORIA MOLTO ANTERIORE. SCRIVEMO $T(m)$ ED $S(m)$ SAREBBERO QUINDI AMBOGLI: OCCHIO DISTINGUERE, PER OGNI M, CASO PEGGIOR, MEDIO E MEGLIO. SE $i \in \text{INPUT}$, INDICHIAMO CON $f(i)$ IL TEMPO CHE L'ALGORITMO IMPREGNA PER ESEGUILO E COM $S(i)$ LA QUANTITÀ DI MEMORIA CHE UTILIZZA DURANTE L'ESECUZIONE SU QUELL'INPUT. SI DISTINGUENO QUINDI 3 FUNZIONI, NON ASSOCIALE IN QUANTO UNIVOCHE:

- $T_{\text{worst}}(m) = \max \{f(i) \mid i \text{ ha dimensione } m\}$
- $T_{\text{best}}(m) = \min \{f(i) \mid i \text{ ha dimensione } m\}$
- $T_{\text{avg}}(m) = \text{AVG} \{f(i) \mid i \text{ ha dimensione } m\}$

PER IL CASO MEDIO, PER OGNI M SI CONSIDERANO TUTTI I POSSIBILI INPUT DI DIMENSIONE M, SIANO i_1, \dots, i_N , E SI FA LA MEDIA ASSIMETRICA DEL TEMPO:

$$T_{\text{avg}}(m) = \frac{f(i_1) + \dots + f(i_N)}{N}$$

SE I POSSIBILI CASI DI INPUT SONO EQUIPROBABILI, MA SI PRESENTANO CON PROBABILITA' p_1, \dots, p_n CON $p_1 + \dots + p_n = 1$ LA MEDIA DEVE ESSERE PESATA: $T_{AVG}(n) = p_1 \cdot t(i_1) + \dots + p_n \cdot t(i_n)$.

POTREMO FARNE 9 POSSIBILI COMBINAZIONI TUTTE BIG-O, AMMAGA E TUTTE MA LE PIU' RILEVANTI SONO DUE:

- $T_{WORST} = O(t(m))$ C'È DEFINITO IL LIMITE SUPERATORIO AL COMPORTAMENTO ASINTOTICO DI UNA FUNZIONE
- $T_{BEST} = \Omega(t(m))$ C'È DEFINITO IL LIMITE INFERIORI AL COMPORTAMENTO ASINTOTICO DI UNA FUNZIONE

SPIEGO POSSEREMO PARLARE DI TUTTA SOLO NEI CASI SOTTOFATI DI UN ALGORITMO E NON NELLA TUTTA DI FATO.

AD ESEMPIO:

INSEGNATORI SONO: CASO PEZZATO $\Theta(m^2)$, CASO MIGLIORE $\Theta(m)$

C'È ANCHE TUTTE OLTREZIONI DA FARE RIGORDO AI VARI CASI:

- 1) IL CASO MIGLIORE SPIEGO NON E' INTERESSANTE
- 2) IL CASO MEDIO SEMPRE INTERESSANTE MA E' DIFFICILE DA CALCOLARE
- 3) IL CASO PEZZATO E' QUELLO PIU' INTERESSANTE

COMPLICATITÀ DI UN PROBLEMA

LA COMPLICATITÀ DI UN PROBLEMA E' DIVISA DA QUELLA DI UN ALGORITMO; SE BASA INFATTI SU TUTTI I POSSIBILI

ALGORITMI POSSIBILI PER QUESTO PROBLEMA. CI SONO DUE ASPECTI PRINCIPALI:

- 1) UN PROBLEMA HA COMPLICATITÀ $O(t(m))$ SE ESISTE UN ALGORITMO CON COMPLICATITÀ $O(t(m))$ CHE LO RISOLVE
- 2) UN PROBLEMA HA COMPLICATITÀ $\Omega(t(m))$ SE TUTTI I POSSIBILI ALGORITMI POSSIBILI HANNO COMPLICATITÀ $\Omega(t(m))$

IL LIMITE INFERIORE PUO' ESSERE QUINDI SOLO TROVATO CON ARGOMENTAZIONI INTUITIVE: PER ESEMPIO LA VERSITA DI M MODI E' IMMEDIATAMENTE CHIARA QUANDO E' $\Omega(m!)$.

TIPI DI PROBLEMA

UN PROBLEMA PUO' ESSERE:

- **CALCOLO QUANTO:**
 - SI CALCOLA UN ALGORITMO ALGOVENTE CON COMPLICATITÀ $O(t(m))$
 - SI E' DIMOSTRATO CHE QUESTO ALGORITMO ALGOVENTE DEVE AVERE $\Omega(t(m))$. (C'È UN PROBLEMA)
- **APPENDE:** SI CONFIRMA CHE ABBIAMO UN ALGORITMO CHE LO RISOLVE, CON COMPLICATITÀ $O(t(m))$, MA NON SI APPRENDE SE STA DAVVERO IL MIGLIORE POSSIBILE. RIMANE QUINDI UN "GAP" E PER COLMARE ABBIAMO DUE MODI:
 - 1) TROVARE UN ALGORITMO PIU' VELOCE CHE RISOLVE IL LIMITE INFERIORE NOTO
 - 2) DIMOSTRARE CHE IL LIMITE INFERIORE E' UNA REALTA' PU' AITO DI QUANTO POSSIAMO

NP - COMPLESSITÀ

CI SONO MOLTI PROBLEMI PER CUI GLI ALGORITMI CONOSCIUTI OCCHI SONO SOLO ESPIRIMENTALI. IN QUESTI CASI SI SOGNA FONDAMENTALE, MA NON SI HA DIMOSTRATO, CHE NON ESISTANO ALGORITMI PIU' EFFICIENTI. QUESTI SONO I COSIDETTI PROBLEMI NP-COMPLESSI: SE TI DANNO ORA UNA SOLUZIONE CONTROLLANDO SE LA SOLUZIONE E' CORRETTA E' FACILE. QUESTO CHE E' DIFFICILE E' TROVARE LA SOLUZIONE.

NP - COMPIESEZZA

Ci sono molti problemi per cui gli algoritmi conosciuti oggi sono solo esponenziali. In questi casi si soffre pesantemente, ma non si
stanno dimostrando, che non esistono algoritmi più efficienti. Questi sono i cosiddetti problemi NP-completi: se ti danno già una
soluzione controllante se la soluzione è corretta è facile. Quello che è difficile è trovare la soluzione. Infine abbiamo i problemi
impossibili.