Fondamenti di Informatica

Prof.ssa E. Gentile

Informatica e Comunicazione Digitale

a.a. 2011-2012

Definizione di Algoritmo

- *Def.*: Per Algoritmo si intende un elenco di istruzioni che specificano una serie di operazioni con le quali è possibile risolvere ogni problema di un dato tipo.

Algoritmo Euclideo

- Dati due numeri positivi a e b, trovare il loro massimo comune divisore.
- Il Massimo Comune Divisore di due (o più) numeri è il numero naturale più grande per il quale possono entrambi essere divisi.

Istruzioni

1. Considerare la coppia di numeri a, b.
2. Confrontare i due numeri considerati (determinare se il primo è uguale, maggiore o minore del secondo).
3. Se i due numeri sono uguali, ognuno di essi fornisce il risultato richiesto; il calcolo si arresta. Altrimenti continua.
4. Se il primo numero è minore del secondo, scambiare i due numeri.
5. Sottrarre il secondo numero dal primo e rimpiazzare i due numeri considerati rispettivamente col sottraendo e con la differenza. Ritornare alla istruzione 2.

Algoritmo numerico

- Un algoritmo per risolvere un sistema di equazioni lineari in due incognite:

$$\begin{cases} a_1 x + b_1 y = c_1 \\ a_2 x + b_2 y = c_2 \end{cases}$$

Soluzioni

$$x = \frac{c_1 b_2 - c_2 b_1}{a_1 b_2 - a_2 b_1}$$

$$y = \frac{a_1 c_2 - a_2 c_1}{a_1 b_2 - a_2 b_1}$$

Soluzioni possibili

- Per qualsiasi scelta dei coefficienti:

$$a_1, a_2, b_1, b_2, c_1, c_2$$

- Ammesso che:

$$a_1 b_2 - a_2 b_1 \neq 0$$

Algoritmo come funzione

- Un algoritmo definisce implicitamente una funzione dall'insieme dei dati di ingresso all'insieme dei dati di uscita. Al tempo stesso indica un procedimento effettivo che permette di determinare, per ogni possibile configurazione in ingresso, i corrispondenti valori in uscita.
- Def.: Dato un algoritmo A, indichiamo con f_A la funzione che associa ad ogni valore in ingresso il corrispondente valore in uscita $f_A(x)$.

Problema

- Def.:
- Un Problema è una funzione $P : D_I \rightarrow D_S$ definita su un insieme D_I di elementi che chiamiamo istanze, ed a valori su un insieme D_S di soluzioni.
- Diciamo che un algoritmo A risolve un problema P se $P(x) = f_A(x)$, per ogni istanza x.

Programma

- «Un programma è l'esposizione di un algoritmo in un linguaggio accuratamente definito. Quindi, il programma di un calcolatore rappresenta un algoritmo, per quanto l'algoritmo stesso sia un costrutto intelligente che esiste indipendentemente da qualsiasi rappresentazione. Allo stesso modo, il concetto di "numero 2" esiste nella nostra mente anche quando non sia espresso graficamente.»

(D.E. Knuth – Fundamental Algorithms vol I, The Art of Computer Programming, 1968)

Costo di calcolo

- Def.: Il costo relativo all'esecuzione di un programma viene definito come la quantità di ***risorse di calcolo*** che il programma utilizza durante l'esecuzione.

Risorse di calcolo

- Le risorse di calcolo a disposizione del programma sono:
  1. Il **Tempo** utilizzato per eseguire l'algoritmo;
  2. Lo **Spazio di lavoro** utilizzato per memorizzare i risultati intermedi; (memoria)
  3. Il **Numero degli esecutori**, se più esecutori collaborano per risolvere lo stesso problema (processori).

Efficienza dell'algoritmo

- Def.: Un algoritmo è efficiente se fa un uso contenuto delle risorse di calcolo a sua disposizione.

Modelli di calcolo

- Def.: Un modello di calcolo è semplicemente una astrazione di un esecutore reale, in cui si omettono dettagli irrilevanti allo studio di un algoritmo per risolvere un problema.

Scelta del modello di calcolo

1. Capacità espressiva del modello in relazione al problema assegnato;
2. Livello di astrazione;
3. Generalità.

Irrisolubilità

- Def.: Un problema è non risolubile algoritmicamente se nessun procedimento di calcolo è in grado di fornire la soluzione in tempo finito.

Problemi risolubili

- Si considera “risolti” una classe di problemi quando si è trovato un algoritmo per risolverli.
- In alcuni casi se non è possibile risolvere problemi di un dato tipo è possibile individuare dei casi particolari nei quali ciò invece è possibile.

Intrattabilità

- Def.: Un problema è intrattabile se qualunque algoritmo che lo risolva richieda una quantità molto elevata di risorse.
- Per esempio, alcuni giochi sono considerati intrattabili

Algoritmi Deterministici

- Un algoritmo deve essere dato sotto forma di una lista finita di istruzioni che specificano il procedimento esatto da eseguire in ogni passo del calcolo. In altri termini, il calcolo non dipende da chi lo esegue; esso è un processo deterministico che può essere ripetuto con successo da chiunque in qualunque momento.

Generalità degli Algoritmi

- Un algoritmo è un'unica lista di istruzioni le quali definiscono il calcolo che può essere eseguito su un insieme qualunque di dati iniziali, e che in ogni caso fornisce il risultato corretto. In altre parole, un algoritmo indica come risolvere non un solo problema particolare, ma un'intera classe di problemi simili.

Definizione di Knuth

- L'Algoritmo è un insieme di regole (o istruzioni) avente le seguenti caratteristiche:
  - Finito
  - Non Ambiguo
  - Dati di ingresso precisi
  - Fornisce un risultato
  - Eseguibile

Algoritmi dei giochi

Prof.ssa E. Gentile

Informatica e Comunicazione Digitale

a.a. 2011-2012

Algoritmi per giochi

- Trovare un algoritmo che fornisca un unico metodo per risolvere ogni particolare problema di una certa classe di problemi simili.

Il gioco dell'undici

- Undici oggetti sono su una tavola. I due giocatori si alternano nel raccogliere 1, 2 o 3 oggetti finché non restano più oggetti sul tavolo.
- Il giocatore costretto a raccogliere l'ultimo oggetto perde.

Strategia vincente

1. A raccoglie 2 oggetti.
2. B raccoglie K oggetti (k ≤ 3)
3. A raccoglie 4-K oggetti

Gioco del pari

- Il gioco comincia con 27 oggetti su una tavola. Alternandosi, i giocatori raccolgono ogni volta da 1 a 4 fiammiferi. Vince chi ha in mano un numero pari di oggetti quando questi sono stati tutti raccolti.

Strategia vincente

1. A raccoglie 2 oggetti
2. Sia r il resto ottenuto dividendo per 6 il numero di oggetti ancora da raccogliere.
3. Se B ha un numero pari di oggetti:
   1. Se r = 2, 3, 4 o 5 allora A raccoglie, rispettivamente 1, 2, 3 o 4 oggetti
4. Se B ha un numero dispari di oggetti:
   1. Se r = 0, 1, 2 o 3 e se sulla tavola restano ancora almeno 4 oggetti, allora A raccoglie, rispettivamente 1, 2, 3 o 4 oggetti
   2. Se r = 4 allora A raccoglie 4 oggetti
   3. Se sulla tavola restano 1 o 3 oggetti allora A li raccoglie tutti

Proprietà dei giochi

1. Il gioco è condotto da due giocatori che eseguono alternativamente una mossa
2. Il gioco termina con esattamente uno di due possibili risultati:
   1. O vince A il giocatore che gioca per primo
   2. O vince B l'avversario
3. Ogni mossa consiste in una scelta da parte del giocatore di una mossa tra un insieme di mosse possibili

Proprietà dei giochi

4. Ad ogni istante del gioco, i giocatori sono informati completamente su tutte le mosse già compiute e su tutte quelle che potranno venir fatte
5. Esiste un limite superiore per il numero di mosse in una partita

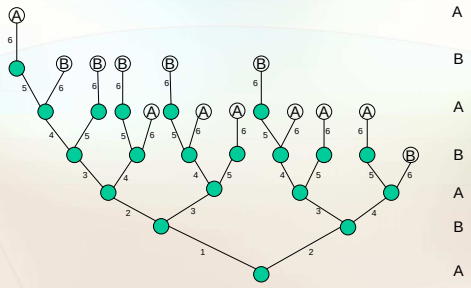
Albero del gioco

- I vertici rappresentano le diverse situazioni che si possono presentare in una partita
- I rami rappresentano le possibili scelte che un giocatore può fare

Gioco del sei

- Su una tavola ci sono 6 oggetti
- Ogni giocatore, a turno, ne sceglie uno o due.
- Perde il giocatore che raccoglie l'ultimo oggetto

Albero del gioco del sei



Regole

- Il massimo livello in un albero viene detto ***ordine***. È uguale alla massima lunghezza di una partita.
- I vertici di livello dispari corrispondono alle mosse di A, quelli di livello pari alle mosse di B.

Teorema: Strategia vincente

- In ogni gioco che soddisfi le proprietà da 1 a 5 già definite, esiste una strategia vincente per uno dei giocatori.

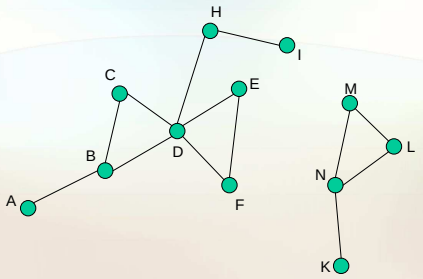
Dimostrazione

- L'algoritmo è costruito per induzione sulla lunghezza *v* della più lunga partita possibile nel gioco (*v* è l'ordine dell'albero)
- *v*=0: nessuna mossa.
- Se è vero per gli ordini ≤*v* dimostriamo che è vero per *v*+1

Labirinto

- Possiamo immaginare un labirinto come un sistema finito di nodi dai quali si dipartono dei corridoi.
- Ogni corridoio congiunge due nodi che vengono detti adiacenti.
- I nodi morti sono quelli da cui si diparte un solo corridoio.

Rappresentazione grafica



Regola

- Diciamo che un nodo Y è ***accessibile*** da un nodo X se esiste un cammino che conduce da X a Y attraverso qualche successione di corridoi intermedi.
- Se Y è accessibile da X allora esiste un cammino semplice tra i due nodi, cioè un cammino che non attraversa più di una volta ciascun nodo.

Algoritmo del labirinto

- Corridoio:
  - Verde: non è mai passato
  - Giallo: è passato una volta
  - Rosso: è passato due volte
- Modalità di passaggio:
  - Svolgendo il filo (passa da verde a giallo)
  - Riavvolgendo il filo (passa da giallo a rosso)
- Non è possibile passare in un corridoio rosso

Condizione dei Nodi

- Arrivo: si è raggiunta la meta
- Ciclo: esistono almeno due altri corridoi gialli che si dipartono dal nodo
- Verde: Esiste almeno un corridoio verde che si diparte dal nodo
- Partenza: Ci troviamo nel nodo iniziale
- Quinto caso: Non si verifica nessuna delle condizioni precedenti

Metodo di ricerca

Condizione	Mossa
1. Arrivo	1. Stop
2. Ciclo	2. Riavvolgere il filo
3. Verde	3. Svolgere il filo
4. Partenza	4. Stop
5. Quinto caso	5. Riavvolgere il filo

Motivazioni del metodo

1. Qualunque sia la posizione di Partenza e di Arrivo nel labirinto, alla fine, dopo un numero finito di mosse, si deve raggiungere un ordine di Stop.
2. Se l'ordine di Stop giunge in Arrivo, il punto è ***accessibile***. Inoltre, siamo giunti attraverso un *cammino semplice* e riavvolgendo il filo possiamo tornare alla Partenza.
3. Se l'ordine di Stop giunge in Partenza, vuol dire che l'Arrivo è ***inaccessibile***.

Esempio 1

Numero della mossa	Condizione trovata	Mossa	Corridoio scelto	Stato dopo la mossa
1	Verde	Svolge	AB	Giallo
2	Verde	Svolge	BC	Giallo
3	Verde	Svolge	CD	Giallo
4	Verde	Svolge	DH	Giallo
5	Verde	Svolge	HI	Giallo
6	Quinto caso	Riavvolge	IH	Rosso
7	Quinto caso	Riavvolge	HD	Rosso
8	Verde	Svolge	DB	Giallo
9	Ciclo	Riavvolge	BD	Rosso
10	Verde	Svolge	DF	Giallo
11	Arrivo	Stop	Stop	Stop

Esempio 2

Numero della mossa	Condizione trovata	Mossa	Corridoio scelto	Stato dopo la mossa
1	Verde	Svolge	KN	Giallo
2	Verde	Svolge	NL	Giallo
3	Verde	Svolge	LM	Giallo
4	Verde	Svolge	MN	Giallo
5	Ciclo	Riavvolge	NM	Rosso
6	Quinto caso	Riavvolge	ML	Rosso
7	Quinto caso	Riavvolge	LN	Rosso
8	Quinto caso	Riavvolge	NK	Rosso
9	Partenza	Stop	Stop	Stop

La regola della destra

- Entra nel labirinto. Quando ti trovi a un nodo prendi il ramo più a destra. Se arrivi a un vicolo cieco, ritorna sui tuoi passi fino all'ultimo nodo e prendi il ramo più a destra tra quelli ancora inesplorati.
- Il modo migliore per visualizzare questo algoritmo è toccare con la mano destra il muro alla propria destra per tutto il labirinto, senza saltare mai un ramo situato alla propria destra.
- Naturalmente la "regola della sinistra" funziona altrettanto bene. È solo necessario essere coerenti una volta entrati nel labirinto.

Il metodo di Tremaux

- Entra nel labirinto. Dapprima vai dove ti pare, contrassegnando il sentiero con un filo, o con sassolini, o molliche di pane, o con qualsiasi cosa tu abbia a disposizione. Continua così finché arrivi:
  - alla meta (se hai fortuna),
  - oppure a un vicolo cieco,
  - oppure a un nodo che avevi già attraversato in precedenza.
- Se arrivi a un vicolo cieco, ritorna al nodo precedente, assicurandoti di contrassegnare il percorso anche a ritroso: in tal modo, se entri ed esci da un vicolo cieco, questo avrà due piste di briciole di pane. Ciò ti permetterà di evitarlo in futuro. Nell'algoritmo di Tremaux non si esplora mai un ramo più di due volte.

Il metodo di Tremaux

- Se arrivi a un nodo già attraversato, fai così:
  - se sei arrivato da un ramo fino ad allora inesplorato (una sola pista di briciole di pane dietro di te) ripercorri quello stesso ramo fino al nodo precedente, altrimenti
  - se c'è un ramo ancora inesplorato a partire dal nodo, prendi questa direzione, altrimenti:
  - prendi qualsiasi ramo che sia stato percorso una sola volta.
- Queste regole esauriscono l'algoritmo di Tremaux. Seguendole scrupolosamente farai un giro completo del labirinto, attraversando ogni ramo due volte, in ciascuna direzione. Ovviamente, puoi anche fermarti quando raggiungi la meta, se non è necessario percorrere l'intero labirinto.

Il metodo di Ore

- Entra nel labirinto. Se non sei già a un nodo, raggiungi quello più vicino. Se non sai quale direzione conduca a quello più vicino, vai a caso fino a incontrare un nodo. Poi contrassegna in qualche maniera questo nodo: sarà la tua casa base.
- Partendo dal nodo base, esplora ogni ramo che si diparte da questo. Metti un contrassegno (ad esempio un ciottolo) all'entrata di ciascun ramo quando cominci a percorrerlo. Esplora ciascun ramo solo fino al nodo successivo. Poi metti un ciottolo all'estremità lontana del ramo e ritorna sui tuoi passi fino alla casa base.

Il metodo di Ore – Prima fase

- Identifica i vicoli ciechi (ad es. con un ciottolo rosso, o chiudendoli con uno spago). Una volta contrassegnato in questo modo, un ramo potrà essere ignorato in futuro. Se un ramo gira su se stesso e ritorna al nodo originario, contrassegnalo come un vicolo cieco: è altrettanto privo di utilità.
- A te interessa individuare quei rami che conducono a nodi con rami nuovi. Alla fine dell'esplorazione preliminare ciascun percorso potenziale verso la meta ha un ciottolo a ciascuna estremità, e tu ti trovi di nuovo al nodo base.
- Adesso esplora fino a una profondità di due nodi. Cammina lungo ciascun ramo che non sia un vicolo cieco fino al nuovo nodo, ed esplora allo stesso modo ciascun ramo che si diparte da questo.

Il metodo di Ore – Seconda fase

- Aggiungi un ciottolo a ciascuna estremità dei rami primari, cosicché adesso avranno due ciottoli su ciascuna estremità, e metti un ciottolo su ciascuna estremità dei nuovi rami secondari. Ciò ti consentirà di ritrovare la strada fino al nodo base: il ramo che conduce a quest'ultimo ha un ciottolo in più rispetto agli altri.
- Come in precedenza, contrassegna le entrate dei vicoli ciechi e dei percorsi circolari. Se un ramo conduce a un nodo già esplorato (con almeno un ciottolo segnaletico) contrassegna anche questo sentiero a entrambe le estremità. Alla fine sarai tornato sui tuoi passi e ti ritroverai di nuovo al nodo base.

Il metodo di Ore – Terza fase

- Nella terza fase di esplorazione, spingiti fino a una distanza di tre nodi dal nodo base, aggiungendo un ciottolo a ciascuna estremità di ogni ramo esplorato.
- Prosegui l'esplorazione sempre più in profondità, fino a raggiungere la meta.
- L'algoritmo di Ore ti permetterà di individuare la via più breve verso la meta. Ovviamente l'andamento dell'esplorazione non seguirà questa via più breve, ma se, ad es., la via più breve attraversa cinque nodi allora la troverai nella quinta fase dell'esplorazione, e saprai che quella è la via più breve.