Aniello Murano

NP-Completezza (seconda parte)

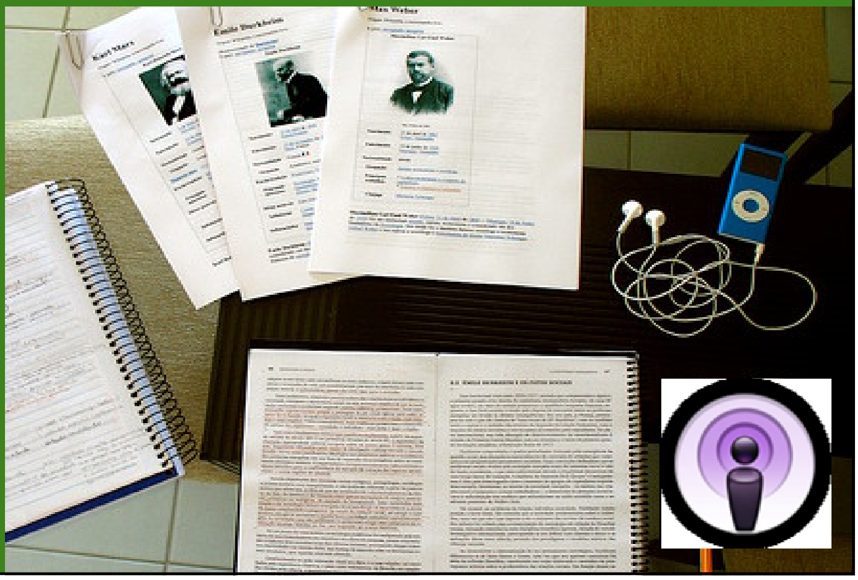
Lezione n.15
Parole chiave:
Np-completezza

Corso di Laurea:
Informatica

Codice:

Email Docente:
murano@ na.infn.it

A.A. 2008-2009



Definizione di NP-COMPLETEZZA

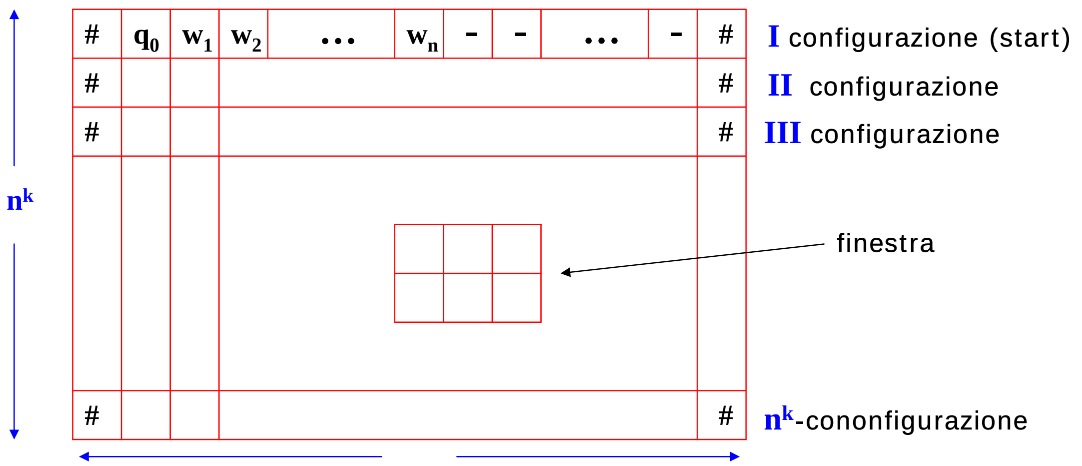
- Si ricordi che un linguaggio B è ***NP-complete*** se soddisfa le seguenti due condizioni:
  1. B è in **NP**,
  2. Ogni linguaggio in **NP** è riducibile in tempo polinomiale a **B**. (hardness)

Teorema di Cook-Levin inizio

- **Teorema:** **SAT** è in NP-complete.
- Dimostrazione:
  - [Appartenenza a NP]: Che **SAT** $\in$ **NP** è ovvio (esercizio per gli studenti).
  - [Hardness]: Dimostriamo adesso che ogni linguaggio A in **NP** è riducibile in tempo polinomiale a **SAT**.
    - Sia A un arbitrario linguaggio tale che A $\in$ **NP.**
    - Sia N la macchina di Turing non deterministica che decide A.
    - Se A $\in$ **NP**, allora il tempo di esecuzione di N è dell'ordine di n^k, dove n è la dimensione dell'input.
    - Mostriamo come trasformare una stringa w in una formula booleana ϕ che "simula" N su input w, nel senso che ϕ è soddisfacibile se e solo se N accetta w.

Teorema di Cook-Levin : Tableaux

- Un tableau per N su w è una tabella $n^k \times n^k$ le cui righe sono le configurazioni di un ramo di calcolo di N su input w = $w_1 \dots w_n$. La prima e ultima colonna contengono solo "#".



- Un tableau è accettante se una delle sue righe è una configurazione accettazione.
- Ogni tableau accettante per N su w corrisponde ad un ramo accettante di N su input W.
- Quindi, determinare se N accetta w equivale a determinare se c'è un tableau accettante per N su w.

#	q_0	w_1	w_2	w_3	…	w_n	-	-	…	-	#
#	w_1	q_7	w_2	w_3	…	w_n	-	-	…	-	#
#	q_5	w_1	$	w_3	…	w_n	-	-	…	-	#

- Sia **C** l'insieme **Q** $\cup\ \Sigma \cup$ **{#}**, dove Q è l'insieme degli stati di N e Σ è l'alfabeto del nastro. **C** è dunque l'insieme di tutti i possibili contenuti delle celle del tableau.
- La cella nella riga i e colonna j è chiamata **cell[i,j]**.. Per ogni cella e per ogni **s**$\in$**C**, si crea una variabile booleana $\mathbf{x}_{i,j,s}$. Il suo significato è "la cell[i,j] contiene il simbolo s".
- La formula Φ si costruisce con queste variabili nel modo seguente:

$$\Phi = \Phi_{\text{cell}} \wedge \Phi_{\text{start}} \wedge \Phi_{\text{move}} \wedge \Phi_{\text{accept}}$$

  - Φ_{cell} asserisce che ciascuna cella contiene esattamente un simbolo
  - Φ_{start} asserisce che la prima riga è la configurazione iniziale su input **w**
  - Φ_{move} asserisce che le righe sono create "legate" le une dalle altre in conformità con la funzione di transizione.
  - Φ_{accept} asserisce che una cella contiene uno stato di accettazione.
  - Φ quindi asserisce che il tableau è un ramo di accettazione della computazione, ovvero che **w** $\in$**A**.

- ϕ_{cell} ci garantisce che ciascuna cella contiene esattamente un simbolo
  - "**cell[i,j]** contiene il simbolo **s**" = $\mathbf{x}_{i,j,s}$
  - "**cell[i,j]** contiene almeno un simbolo " = $\left(\bigvee_{s\in C} x_{i,j,s}\right)$
  - "**cell[i,j]** non contiene sia **s** e **t**" = $(\underline{x}_{i,j,s} \vee \underline{x}_{i,j,t})$
  - "**cell[i,j]** contiene al più un simbolo" = $\left(\bigwedge_{\substack{s,t\in C\\ s\neq t}} (\underline{x}_{i,j,s} \vee \underline{x}_{i,j,t})\right)$
  - Ciascuna "**cell[i,j]** contiene esattamente un simbolo"

$$\underbrace{\overbrace{\bigwedge_{1\le i,j\le n^k}}\ \overbrace{\left[\left(\bigvee_{s\in C} x_{i,j,s}\right) \wedge \left(\bigwedge_{\substack{s,t\in C\\ s\neq t}} (\underline{x}_{i,j,s} \vee \underline{x}_{i,j,t})\right)\right]}}_{\phi_{\text{cell}}}$$

- ϕ_{start} asserisce che la prima riga è la configurazione iniziale su input w

#	q_0	w_1	w_2	...	w_n	-	-	...	-	#

I configurazione (start)

"**cell[1,1]** contiene **#**" = $x_{1,1,\#}$

"**cell[1,2]** contiene q_0" = $x_{1,2,q_0}$

...

$$\phi_{start} = \begin{cases} x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \\ x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \dots \wedge x_{1,n+2,w_n} \wedge \\ x_{1,n+3,-} \wedge x_{1,n^k-1,-} \wedge x_{1,n^k,\#} \end{cases}$$

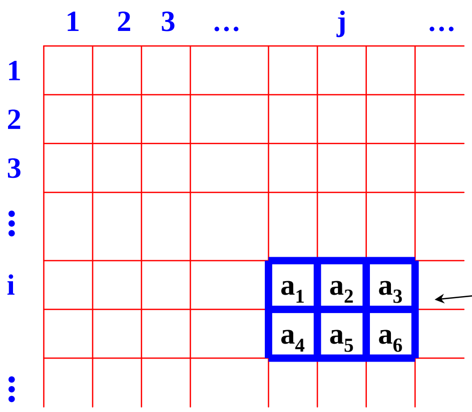
ϕ_{accept} asserisce che una delle celle contiene lo stato di accept

"**cell[i,j]** contiene q_{accept}" = $x_{i,j,q_{accept}}$

$$\phi_{accept} = \bigvee_{1 \le i,j \le n^k} x_{i,j,q_{accept}}$$

Nota: Il controllo che ogni riga del tableau contiene esattamente un solo stato è garantita dalla formula ϕ_{move} definita nelle prossime diapositive

Teorema di Cook-Levin : finestre (windws)



la finestra (i,j).
Si noti che cell[i,j]=a_2

Il contenuto di una finestra è ***legale*** se il suo contenuto può apparire in un tableau "possibile" per **N**.

Teorema di Cook-Levin : esempi di finestre legali e non

Sia δ la funzione di transizione di **N**, assumiamo di avere
$\delta(q_1,a) = \{(q_1,b,R)\}$ e $\delta(q_1,b) = \{(q_2,c,L),(q_2,a,R)\}$.
Le seguenti finestre sono legali o illegali?

a	q_1	b
q_2	a	c

a	q_1	b
a	a	q_2

a	a	q_1
a	a	b

#	b	a
#	b	a

a	b	a
a	b	q_2

b	b	b
c	b	b

a	b	a
a	a	a

a	q_1	b
q_1	a	a

b	q_1	b
q_2	b	q_2

Il teorema di Cook-Levin: Domanda sulle Windows

- **Claim**: Se la riga superiore del tableau è la configurazione iniziale e tutte le finestre nel tableau sono legali, allora ogni riga del tableau è una configurazione che legalmente segue quella precedente.
- **Dimostrazione:**
  - Si prendano in esame ogni due righe adiacenti (configurazioni).
  - Nella configurazione superiore, ogni cella che non è adiacente al simbolo di uno stato e non contiene il simbolo di limite # è il simbolo di una cella in una finestra di tre celle la cui parte centrale non è interessata da transizioni di N.
  - Pertanto quel simbolo centrale, in una finestra legale, non deve cambiare nella parte inferiore della finestra.

x	b	y
?	b	?

  - La finestra superiore che invece contiene nella parte centrale uno stato, sicuramente sarà interessato da una transizione.
  - Le corrispondenti tre posizioni nella parte inferiore saranno aggiornate in base alla funzione di transizione.
  - Pertanto, se la configurazione superiore è una configurazione legale, allora è una configurazione legale anche quella inferiore

	q_1	

Teorema di Cook-Levin : ϕ_{move}

- Φ_{move} afferma che le righe sono legate le une alle altre in accordo alla funzione di transizione.
- Una 6-tupla (a_1, a_2, a_3, a_4, a_5, a_6) di simboli da C è una finestra legale se rispetta le proprietà descritte precedentemente.
- Si noti che il numero di 6-tuple legali è fisso e non dipende da w.
- Il numero massimo di 6-tuple legali è **$|C|^6$**.

a_1	a_2	a_3
a_4	a_5	a_6

$$\underbrace{\bigvee_{\substack{(a_1,\ldots,a_6)\\ \text{is legal}}} \overbrace{\left(x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6}\right)}^{\text{“il contenuto della finestra la cui prima riga è centrata su } \mathbf{cell[i,j]} \text{ è } (a_1,\ldots,a_6)\text{”}}}_{\text{“la } \mathbf{finestra\ (i,j)} \text{ è legale”}}$$

$$\phi_{move} = \bigwedge_{\substack{1 \le i \le n^k-1\\ 2 \le j \le n^k}} (\text{la finestra } (i,j) \text{ è legale})$$

- La nostra riduzione costruisce ϕ e la sua complessità di tempo è asintoticamente la stessa della dimensione di ϕ.
- E' importante verificare che questa dimensione è polinomiale nella dimensione n macchina di Turing N.
- Per la costruzione data, è sufficiente verificare la polinomialità (in n) dei quattro congiunzioni di ϕ.

Qual'è la taglia di $\phi_{\mathbf{start}}$?

Qual'è la taglia di $\phi_{\mathbf{accept}}$?

Qual'è la taglia di $\phi_{\mathbf{cell}}$?

Qual'è la taglia di $\phi_{\mathbf{move}}$?

La complessità della riduzione risulta essere **$O(n^{2k})$**, dunque la riduzione è **polinomiale**.

Nota: Se invece delle finestre avessimo preso in considerazione due intere configurazioni, allora la complessità di move sarebbe stata di C^{n^k}, dunque esponenziale