Progetto e analisi di algoritmi

Roberto Cordone
DTI - Università degli Studi di Milano
Polo Didattico e di Ricerca di Crema

Tel. 0373 / 898**089**
E-mail: **`cordone@dti.unimi.it`**
Ricevimento: **su appuntamento**
Web page: **`http://www.dti.unimi.it/~cordone`**

Lezioni: **Martedì dalle 11.00 alle 13.00**
Giovedì dalle 11.00 alle 13.00

Progetto per riduzione

1. Si trova una riduzione R del problema P al problema Q

$$P \preceq_R Q \qquad (I_P \rightarrow I_Q \text{ per ogni } I_P \in P)$$

2. Si risolve Q con un algoritmo A_Q

$$I_Q \rightarrow S_Q$$

3. Si trasforma la soluzione dell'istanza di Q nella soluzione dell'istanza di P

$$S_Q \rightarrow S_P$$

Se i tre passaggi sono polinomiali, l'algoritmo A_P è polinomiale

Riduzione da un problema a sé stesso

Il problema Q può essere un sottoinsieme del problema P

$$SAT \preceq_T 3 - SAT$$

Ogni problema P ha un sottoinsieme $P_0 \subseteq P$ di istanze semplici

Idea: si può ridurre P a P_0?

Di solito no, ma spesso si può ridurre $P_1 \subset P$ a P_0…

$$P \not\preceq_R P_0 \;\text{ ma }\; \exists P_1 \subset P : P_1 \preceq_R P_0$$

…e si può ripetere il procedimento…

$$P \not\preceq_R P_1 \text{ ma } \exists P_2 \subset P : P_2 \preceq_R P_1$$

Un esempio: l'ordinamento

- P_0: è banale ordinare un vettore di 0 elementi
- P_1: se ho un vettore di 1 elemento, posso toglierne uno, ordinare il vettore residuo di 0 elementi (istanza di P_0) e reinserire l'elemento in modo ordinato
- P_2: se ho un vettore di 2 elementi, posso toglierne uno, ordinare il vettore residuo di 1 elemento (istanza di P_1) e reinserire l'elemento in modo ordinato
- $\ldots$
- P_n: se ho un vettore di n elementi, posso toglierne uno, ordinare il vettore residuo di $n-1$ elementi (istanza di P_{n-1}) e reinserire l'elemento in modo ordinato

Progetto per induzione

1. Partiziono le istanze in “classi” associate ai numeri naturali
2. Considero l’istanza corrente:
   (a) se è di classe 0, la risolvo con un algoritmo ad hoc
   (b) se è di classe > 0, la riduco a un problema di classe inferiore e torno al punto 2
3. Trasformo la soluzione del problema ridotto in quella del problema iniziale

La “classe” non è la “dimensione”, anche se spesso sono legate

Come sappiamo che l’algoritmo è corretto,

cioè che funziona per tutte le istanze?

Induzione e numeri naturali

Ripartiamo le istanze di P in una successione di sottoinsiemi P_k

$$P = \bigcup_{k=0}^{+\infty} P_k \qquad P_i \cap P_j \text{ per } i \neq j$$

Ora dobbiamo dimostrare che

l'algoritmo A risolve le istanze di P_k, per ogni $k \in \mathbb{N}$

Per farlo, sfruttiamo la definizione stessa dei numeri naturali

Assiomi di Peano

L'insieme dei numeri naturali $\mathbb{N}$ è definito da:

1. $0 \in \mathbb{N}$ (l'insieme non è vuoto)
2. La funzione $S : \mathbb{N} \to \mathbb{N}$ definisce $\forall n \in \mathbb{N}$ uno e un solo $n' = S(n) \in \mathbb{N} : n' \neq n$ (ogni numero ha un successore)
3. $\nexists n \in \mathbb{N} : 0 = S(n)$ (0 non è successore di nessun numero)
4. $n \neq n' \Rightarrow S(n) \neq S(n')$ (a numeri diversi, successori diversi)
5. principio di induzione

$$\begin{cases} K \subseteq \mathbb{N} \\ 0 \in K \\ n \in K \Rightarrow S(n) \in K \end{cases} \Rightarrow K = \mathbb{N}$$

Applicazione agli algoritmi

Sia K il sottoinsieme dei numeri naturali tali che l'algoritmo A risolve tutte le istanze di P_k per ogni $k \in K$

Supponiamo che

1. A risolve tutte le istanze in P_0, cioè $0 \in K$
2. Se A risolve tutte le istanze in P_k, allora risolve tutte le istanze in P_{k+1}, cioè $k \in K \Rightarrow k+1 \in K$

Allora $K = \mathbb{N}$, cioè per ogni $k \in \mathbb{N}$ A risolve P_k

Quindi A risolve P