Tipi di problemi

- Decidibili o indecidibili
- Trattabili o intrattabili.

Decidibili/indecidibili

- Decidibile: l'algoritmo termina e produce la soluzione corrispondente in tempo finito.

- Indecidibile: la soluzione è calcolabile in tempo finito solo per alcune istanze dei dati in ingresso.

Un esempio di problema indecidibile è il problema della terminazione.

Prendiamo un algoritmo H che prende in ingresso un algoritmo A e una istanza i dei suoi dati in ingresso, e restituisce in tempo finito un valore di verità H(A, i) che indica se A termina su i.

Consideriamo un secondo algoritmo H' che prende in ingresso A e termina solo se H(A, A) è falso.

H'(H') termina solo se H(H', H') è falso. Ciò è assurdo, dunque H non può esistere e il problema della terminazione è indecidibile.

Trattabili / Intrattabili

Possibilità di risolvere problemi decidibili in maniera efficiente.

- Un problema decidibile è intrattabile se

non è risolvibile in tempo polinomiale nemmeno da un algoritmo non deterministico.

- P: problemi risolvibili da un algoritmo deterministico in tempo polinomiale.
- NP: risolvibili in tempo polinomiale da un algoritmo non deterministico.

P ⊆ NP.

Problema della soddisfacibilità (SAT)

Data un'espressione logica in forma

normale congiuntiva, stabilire se esiste un assegnamento dei suoi valori che la rende vera.

- Non conosciamo nessun algoritmo che lo risolve, ma possiamo determinare in maniera efficiente se una soluzione è tale oppure no.

Riducibilità

Intesa in tempo polinomiale.

- La soluzione di ogni istanza di P_1 può essere ottenuta risolvendo la corrispondente istanza di P_2, calcolabile in t.p.

Dati due problemi di decisione P_1 e P_2, si dice che P_1 è riducibile in t.p. a P_2 ($P_1 \rightarrow P_2$ o $P_1 \leq P_2$) se esiste un algoritmo deterministico polinomiale che calcola una funzione f tale che:

- $f: I_1 \longrightarrow I_2$, ovvero trasforma l'input di P_1 in un input di P_2
- $\forall i \in I_1, \forall s \in \{0,1\}$:
  $(i,s) \in P_1 \longleftrightarrow (f(i),s) \in P_2$

Ovvero la soluzione al problema è la stessa usando come input on i e la sua trasformazione.

Conseguenze:

- Se $B \in P$ e $A \rightarrow B$, allora $A \in P$
- Se A non può essere risolto in t.P., allora B non può essere risolto in t.P.
- La difficoltà del problema A viene trasferita su B.

P vs NP

Appare naturale chiedersi quali problemi sono più difficili in NP.

Si cerca un problema B tale che ogni altro problema in NP sia riducibile a B.

- Esiste tale problema? Si.

Teorema di Cook (1971): dimostra che ogni problema di classe NP può essere ricondotto in tempo polinomiale al problema della soddisfacibilità booleana

(SAT).

Corollario: se esiste un algoritmo che risolve SAT in t.p., allora P = NP.

Un problema in NP è detto NP-completo se:

- SAT è riducibile ad esso in tempo polinomiale

Se esiste un algoritmo polinomiale che risolve un problema NP-completo, siccome questi problemi sono tutti riducibili uno all'altro, esiste un algoritmo polinomiale per ogni altro problema e quindi P = NP.