Problemi NP-completi

- Sono i problemi più difficili all'interno della classe NP
  - Se esistesse un algoritmo polinomiale per risolvere uno solo di questi problemi, allora
    - tutti i problemi in NP potrebbero essere risolti in tempo polinomiale,
    - dunque P = NP
  - Quindi:
    - tutti i problemi NP-completi sono risolvibili in tempo polinomiale oppure nessuno lo è

Riduzioni polinomiali

Π_1 e Π_2 = problemi decisionali

I_1 e I_2 = insiemi delle istanze di input di Π_1 e Π_2

Π_1 *si riduce in tempo polinomiale a* Π_2

$$\Pi_1 \leq_p \Pi_2$$

se esiste una *funzione* $f: I_1 \rightarrow I_2$ calcolabile in tempo polinomiale tale che, per ogni istanza x di Π_1

x è un'istanza accettabile di Π_1

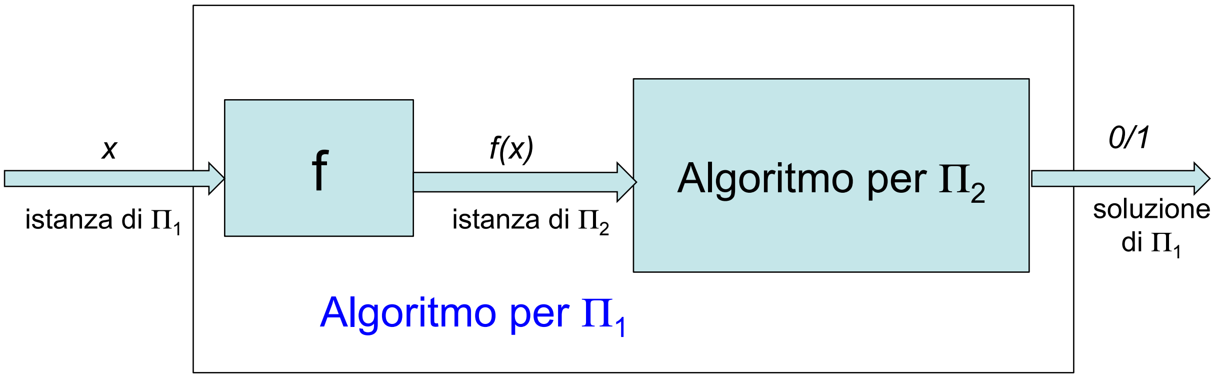
SE E SOLO SE

f(x) è un'istanza accettabile di Π_2

Riduzioni polinomiali

Se esistesse un algoritmo per risolvere Π_2 potremmo utilizzarlo per risolvere Π_1

$$\Pi_1 \leq_p \Pi_2 \quad e \quad \Pi_2 \in P \quad \Rightarrow \quad \Pi_1 \in P$$



Problemi NP ardui

Un problema decisionale Π si dice NP-arduo se

$$\text{per ogni } \Pi' \in NP,\ \Pi' \leq_p \Pi$$

Problemi NP completi

Un problema decisionale Π si dice NP-completo se

$\Pi \in NP$

Π è NP-arduo

Problemi NP completi

- Dimostrare che un problema è in NP può essere facile
  - Esibire un certificato polinomiale
- Non è altrettanto facile dimostrare che un problema Π è NP-arduo
  - Bisogna dimostrare che TUTTI i problemi in NP si riducono polinomialmente a Π
  - In realtà la prima dimostrazione di NP-completezza aggira il problema

SAT

Soddisfacibilità di formule booleane

Definizioni

- Insieme V di variabili Booleane
  - Letterale: variabile o sua negazione
  - Clausola: disgiunzione (OR) di letterali
- Un'espressione Booleana su V si dice in forma normale congiuntiva (FNC) se è espressa come congiunzione di clausole (AND di OR)

Esempio

$$V = \{x, y, z, w\}$$

$$FNC: (x \vee \overline{y} \vee z) \wedge (\overline{x} \vee w) \wedge y$$

SAT

- Data una espressione in *forma normale congiuntiva*

  verificare se esiste una assegnazione di valori di verità alle variabili che rende l'espressione vera

Esempio

- La formula

$$(x \vee \overline{y} \vee z) \wedge (\overline{x} \vee w) \wedge y$$

è soddisfatta dall'assegnazione

$$x = 1 \quad y = 1 \quad z = 0 \quad w = 1$$

SAT ∈ NP

Certificato per SAT?

Un'assegnazione di valori (0 o 1) alle variabili che renda vera l'espressione

Teorema di Cook

SAT

problema della soddisfacibilità di una espressione booleane in forma normale congiuntiva (FNC)

Teorema

SAT è NP completo

Teorema di Cook (idea)

- *Cook ha mostrato un algoritmo che*

  dati un qualunque problema Π ed una qualunque istanza x per Π

  costruisce una espressione Booleana in forma normale congiuntiva che descrive il calcolo di un algoritmo per risolvere Π su x

- *L'espressione è vera se e solo se l'algoritmo restituisce 1*

Problemi NP completi

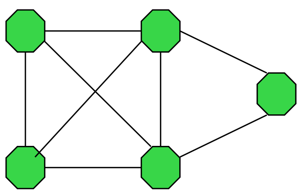
Un problema decisionale Π è NP completo se

- $\Pi \in NP$
- SAT $\leq_p \Pi$

  (o un qualsiasi altro problema NPC)

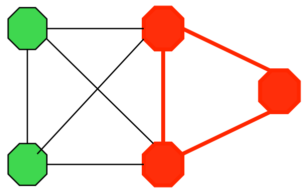
Riduzione: SAT $\leq_p$CLIQUE

- Dato un grafo G = (V,E) e un intero k > 0, stabilire se G contiene una clique di k nodi



CLIQUE

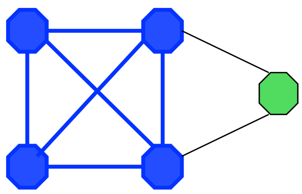
- Dato un grafo G = (V,E) e un intero k > 0, stabilire se G contiene una clique di k nodi



Clique di 3 nodi

CLIQUE

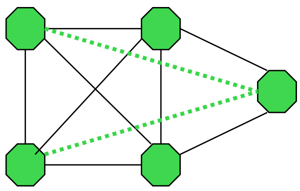
- Dato un grafo G = (V,E) e un intero k > 0, stabilire se G contiene una clique di k nodi



Clique di 4 nodi

CLIQUE

- Dato un grafo G = (V,E) e un intero k > 0, stabilire se G contiene una clique di k nodi



Non contiene clique di 5 nodi

CLIQUE è NP completo

SAT $\leq_p$ CLIQUE

data un'espressione booleana F in forma normale congiuntiva con k clausole

costruire in tempo polinomiale

un grafo G che contiene una **clique di k vertici se e solo se F è soddisfacibile**.

Riduzione: vertici

- Ad ogni letterale in ciascuna clausola di F corrisponde un vertice in G.

Esempio:

$F = (a \vee b) \wedge (!a \vee !b \vee c) \wedge !c$

$G = (V, E)$,

$V = \{ a^1, b^1, !a^2, !b^2, c^2, !c^3 \}$

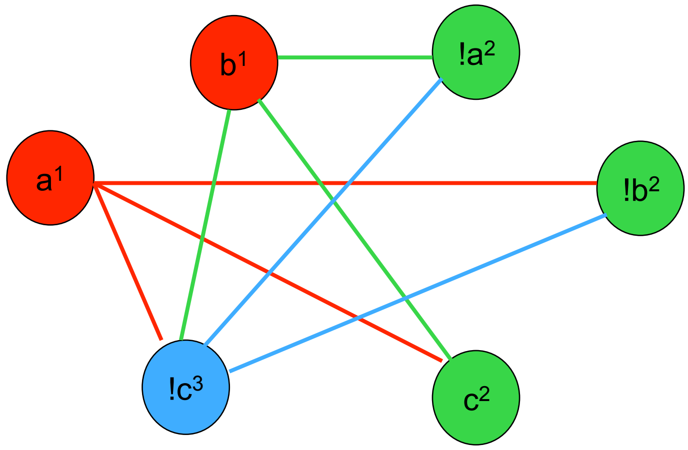
Riduzione: archi

$$(x^i , y^j) \in E \iff i \neq j \;\; e \;\; x \neq !y$$

Due letterali sono adiacenti in G se e solo se

- *appartengono a clausole diverse*
- *possono essere veri contemporaneamente.*

F = (a ∨ b) ∧ (!a ∨ !b ∨ c) ∧ !c



Clique in G

- *composta da k nodi*

  uno per ogni clausola di F

- *non può contenere due nodi della stessa clausola*

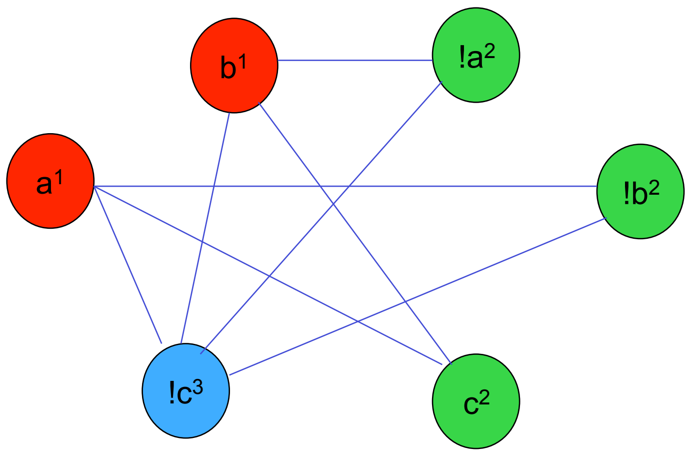
  perché non sono adiacenti.

Riduzione

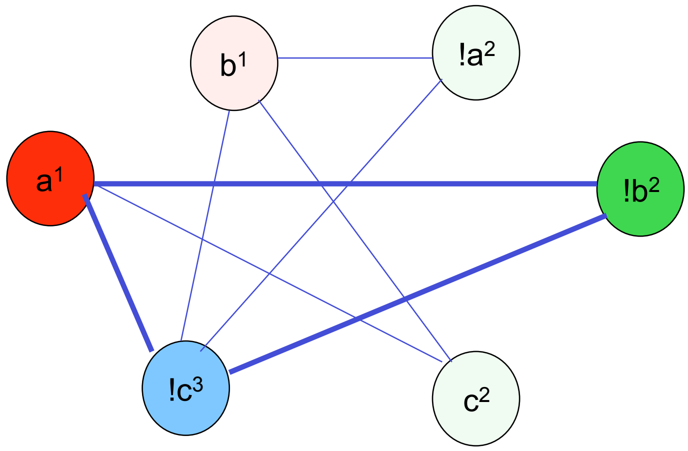
G contiene una clique ⇒ F è soddisfacibile

- si dà valore **1** (true) ai k letterali che corrispondono ai nodi della clique
- tutte la clausole corrispondenti diventano di valore **1** (true)
- F = **1** (true), soddisfacibile.

$F = (a \lor b) \land (!a \lor !b \lor c) \land !c$

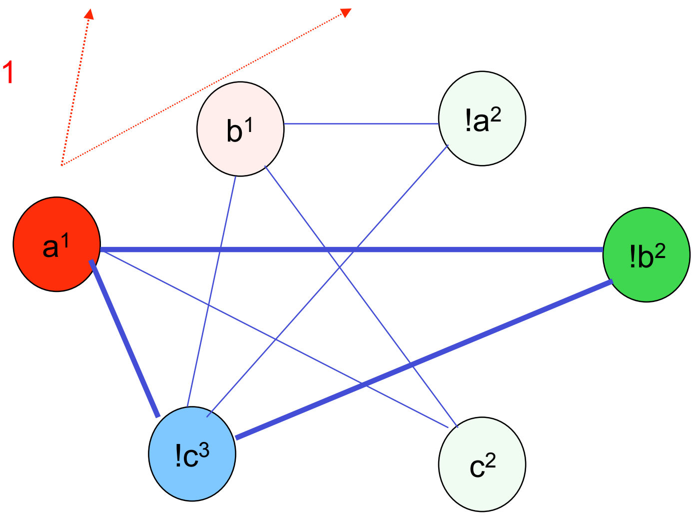


F = (a ∨ b) ∧ (!a ∨ !b ∨ c) ∧ !c

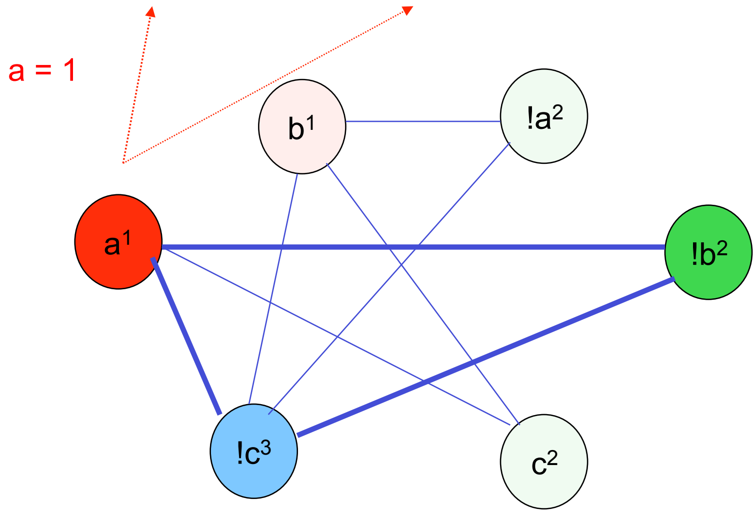


F = (**a** ∨ b) ∧ (**!a** ∨ !b ∨ c) ∧ !c

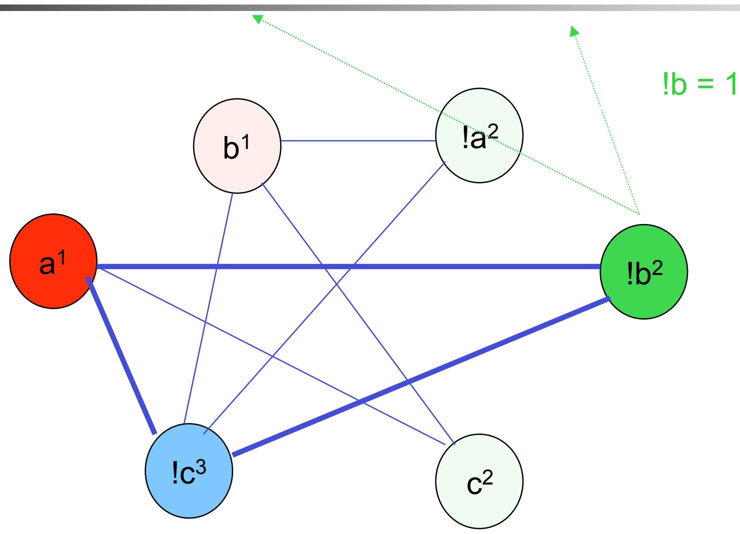
a = 1



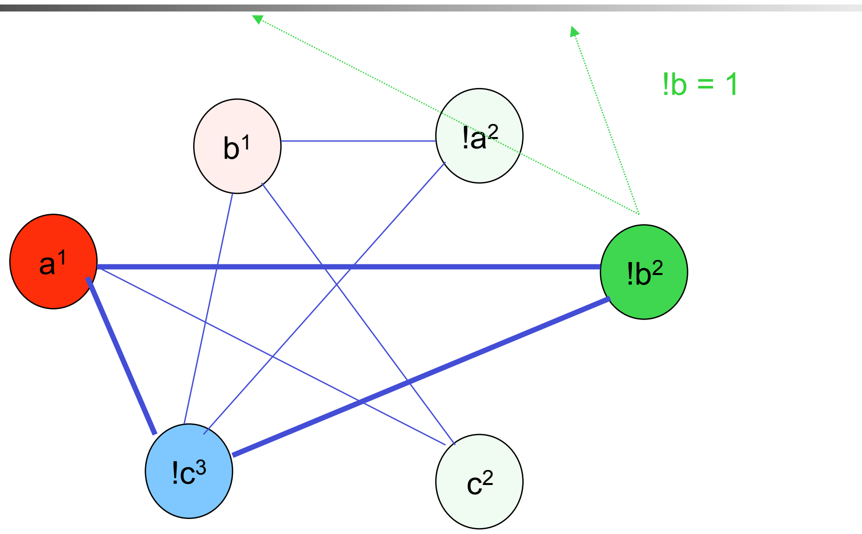
$F = (\mathbf{1} \vee b) \wedge (0 \vee !b \vee c) \wedge !c$



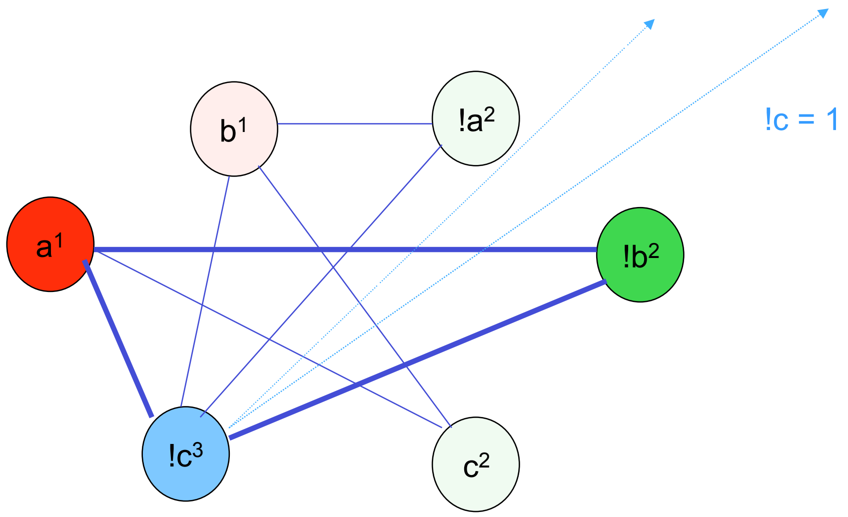
$F = (\mathbf{1} \vee \mathbf{b}) \wedge (0 \vee \mathbf{!b} \vee c) \wedge !c$



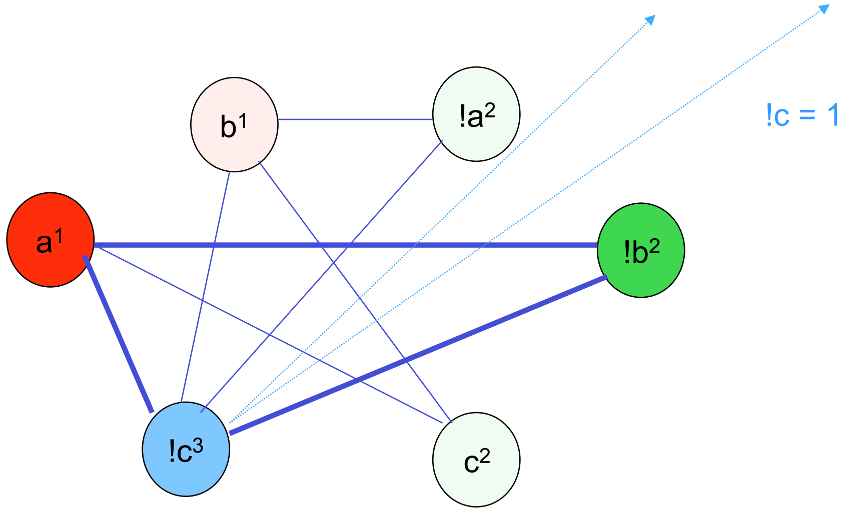
$F = (1 \lor 0) \land (0 \lor 1 \lor c) \land !c$



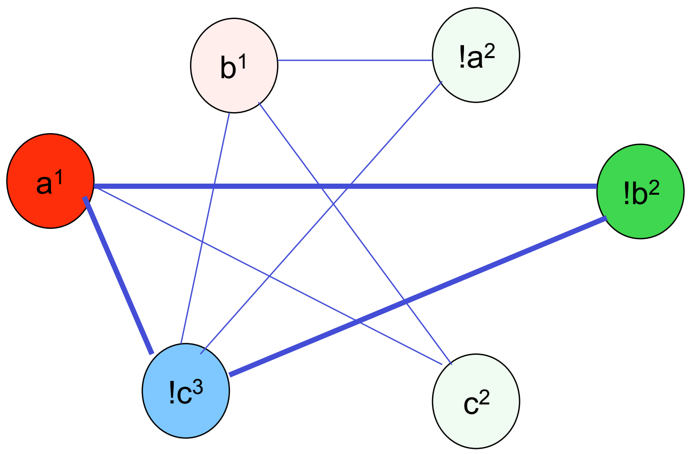
$F = (1 \lor 0) \land (0 \lor 1 \lor c) \land !c$



$F = (1 \lor 0) \land (0 \lor 1 \lor 0) \land 1$



$F = (1) \land (1) \land 1 = 1$



Riduzione

F è soddisfacibile $\Rightarrow$ G contiene una clique

- esiste almeno un letterale vero per ogni clausola
- i corrispondenti vertici in G formano una clique.

Riduzione

- *La riduzione da* F *a* G = (V,E) *si esegue in tempo polinomiale:*
  - n = # variabili
  - k = # clausole
  - $|V| \leq n\ k$
  - l'esistenza di un arco si stabilisce in tempo costante
  - $|E| \leq O((n\ k)^2)$

Problemi NP equivalenti

- *SAT* $\leq_p$ *CLIQUE* $\Rightarrow$ CLIQUE è NP completo
- SAT è NP completo $\Rightarrow$ *CLIQUE* $\leq_p$ *SAT*
- *SAT e CLIQUE sono* ***NP equivalenti***.
- *Tutti i problemi NP completi sono tra loro* ***NP equivalenti***.

Gerarchia delle classi