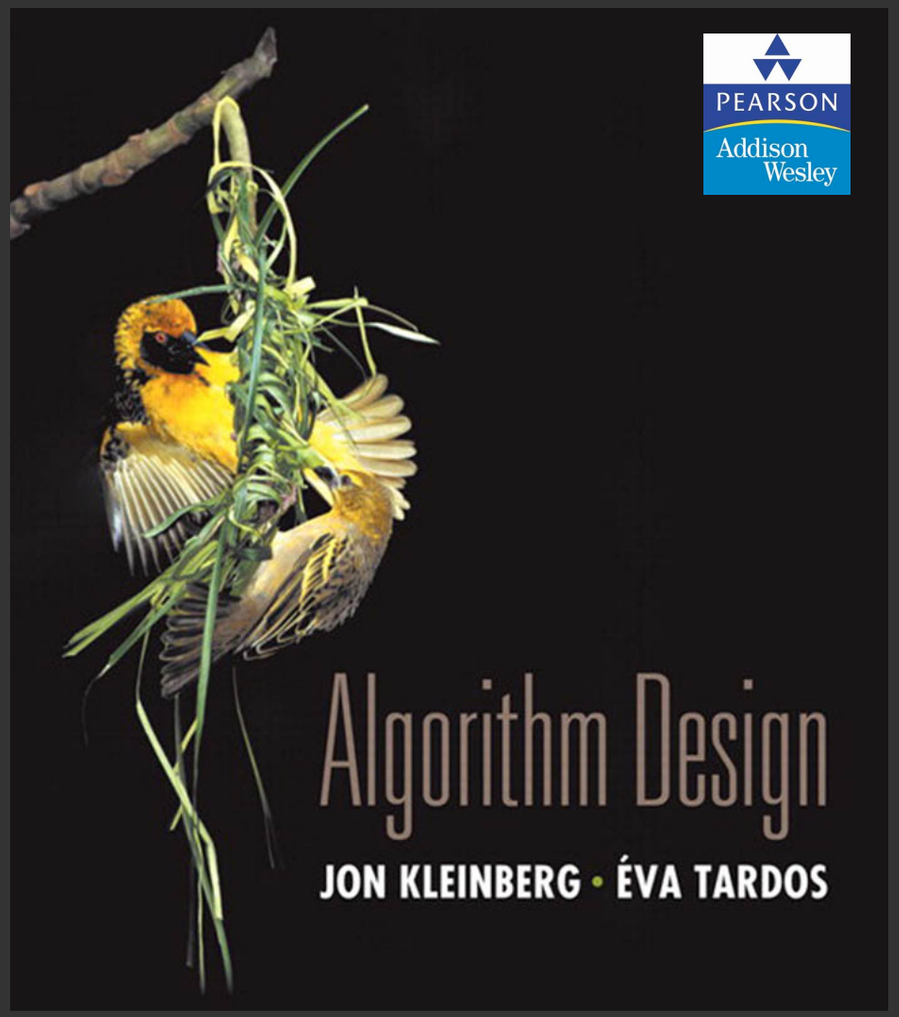


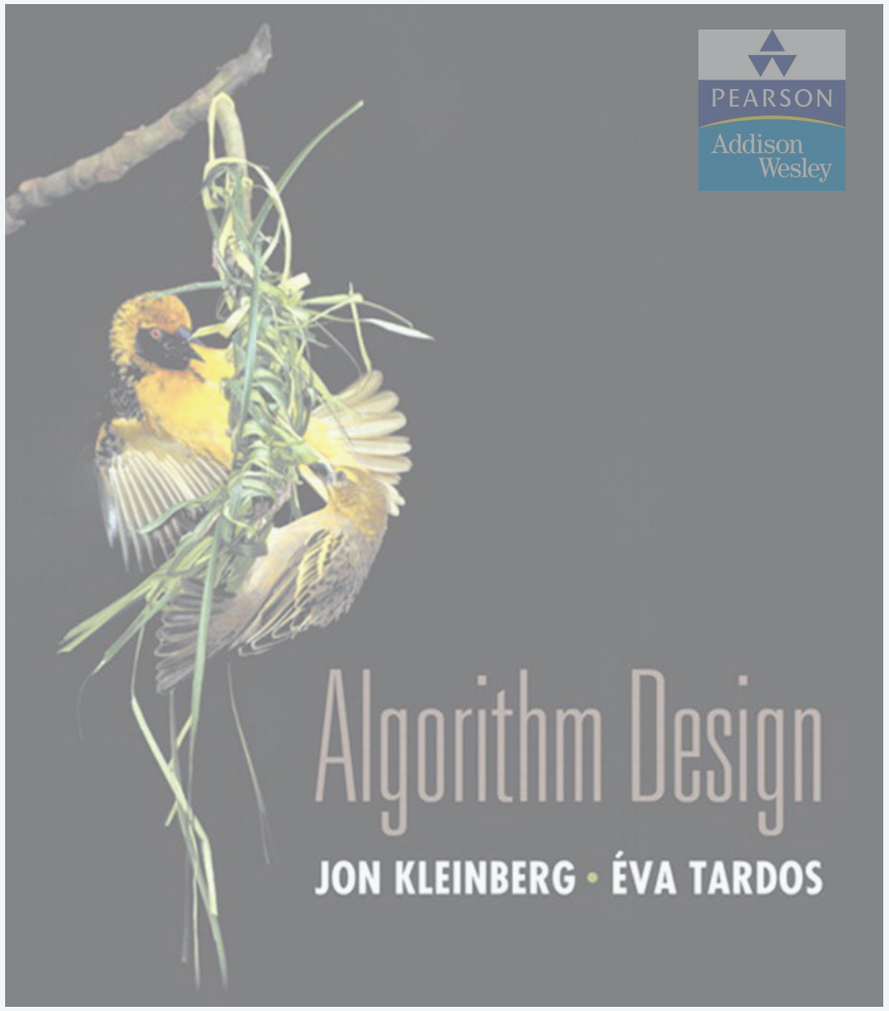
Traduzione e adattamento di Vincenzo Bonifaci
Original lecture slides by Kevin Wayne

http://www.cs.princeton.edu/~wayne/kleinberg-tardos

8. Intrattabilità I

- *riduzioni tempo-polinomiali*
- *problemi di copertura e impaccamento*
- *problemi di soddisfacimento di vincoli*
- *problemi di sequenziamento*
- *problemi di partizionamento*
- *colorazione di grafi*
- *problemi numerici*

Last updated on 10/06/2020 11:46



SECTION 8.1

8. INTRATTABILITÀ I

- *riduzioni tempo-polinomiali*
- *packing and covering problems*
- *constraint satisfaction problems*
- *sequencing problems*
- *partitioning problems*
- *graph coloring*
- *numerical problems*

Paradigmi di progettazione algoritmica e "antiparadigmi"

Paradigmi di progettazione algoritmica.

- Avido.
- Divide et impera.
- Programmazione dinamica.
- Dualità.
- Riduzioni.
- Ricerca locale.
- Randomizzazione.

"Antiparadigmi" di progettazione algoritmica.

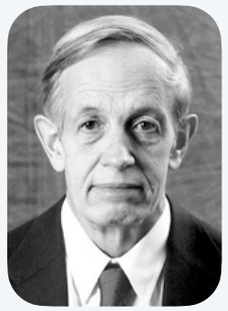
- **NP**-completezza. Algoritmo $O(n^k)$ improbabile.
- **PSPACE**-completezza. Algoritmo di certificazione $O(n^k)$ improbabile.
- Indecidibilità. Nessun algoritmo può esistere per il problema.

Classificazione di problemi in base ai loro requisiti computazionali

D. Quali problemi saremo in grado di risolvere in pratica?

Una definizione operativa. Quelli con algoritmi tempo-polinomiali.

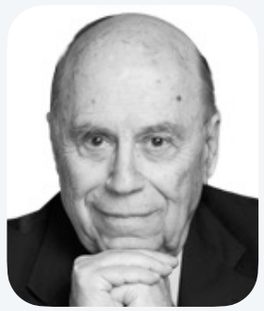

von Neumann (1953)


Nash (1955)


Gödel (1956)


Cobham (1964)


Edmonds (1965)


Rabin (1966)

macchina di Turing, word RAM, circuiti uniformi, ...

Teoria. La definizione è ampia e robusta.

le costanti tendono ad essere piccole, p.e., $3n^2$

Pratica. Algoritmi tempo-polinomiali scalano su istanze enormi.

Classificazione di problemi in base ai loro requisiti computazionali

D. Quali problemi saremo in grado di risolvere in pratica?

Una definizione operativa. Quelli con algoritmi tempo-polinomiali.

sì	probabilmente no
cammino minimo	cammino massimo
taglio minimo	taglio massimo
2-satisfiability	3-satisfiability
4-colorabilità planare	3-colorabilità planare
vertex cover bipartito	vertex cover
abbinamento	abbinamento 3D
test di primalità	fattorizzazione
ottimizzazione lineare	ottimizzazione lineare intera

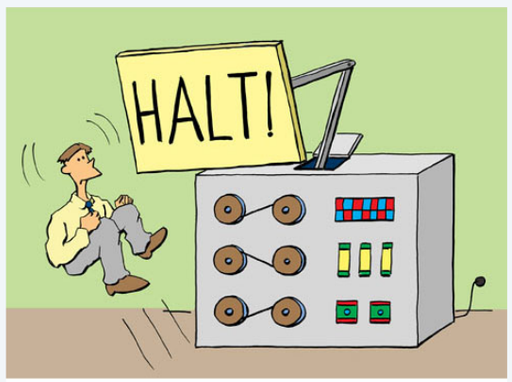
Classificazione di problemi

Desiderata. Classificare problemi in quelli che possono essere risolti in tempo polinomiale e quelli che non possono.

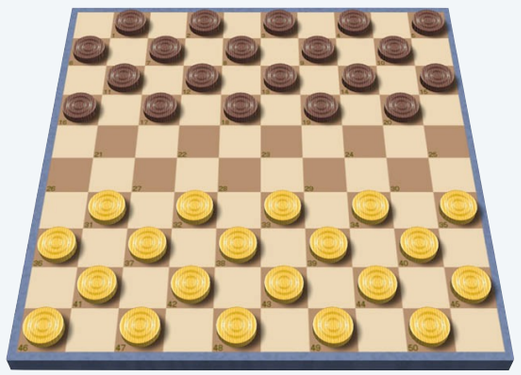
Problemi che dimostrabilmente richiedono tempo esponenziale. (taglia dell'input $= c + \log k$)

- Dato un programma di taglia costante, esso termina in al più k passi?
- Data una configurazione di una generalizzazione n-per-n della dama, il nero può garantirsi di vincere?

(con la regola della cattura forzata)



Alan designed the perfect computer



Notizia frustrante. Un enorme numero di problemi fondamentali ha resistito la classificazione per decenni.

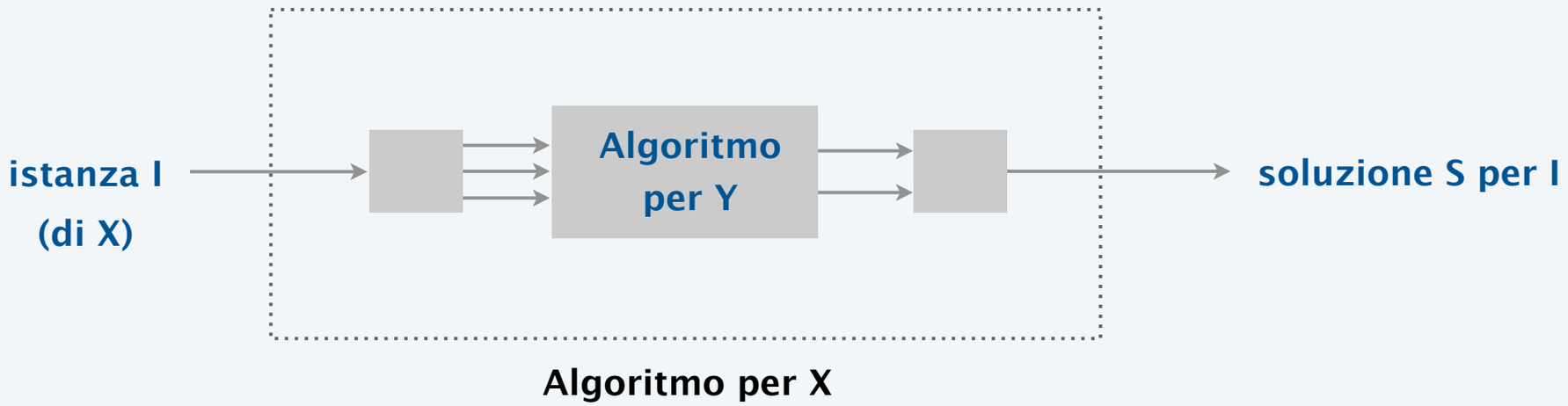
Riduzioni tempo-polinomiali

Desiderata'. Supponiamo di poter risolvere il problema Y in tempo polinomiale. Cos'altro potremmo risolvere in tempo polinomiale?

Riduzione. Problema X (Cook-)riduce in tempo-polinomiale al problema Y se istanze arbitrarie del problema X possono essere risolte attraverso:

- Un numero polinomiale di normali passi di calcolo, più
- Un numero polinomiale di invocazioni ad un oracolo che risolve il problema Y.

modello computazionale con l'aggiunta di uno speciale hardware che risolve istanze di Y in un singolo passo



Riduzioni tempo-polinomiali

Desiderata'. Supponiamo di poter risolvere il problema Y in tempo polinomiale. Cos'altro potremmo risolvere in tempo polinomiale?

Riduzione. Problema X (Cook-)riduce in tempo-polinomiale al problema Y se istanze arbitrarie del problema X possono essere risolte attraverso:

- Un numero polinomiale di normali passi di calcolo, più
- Un numero polinomiale di invocazioni ad un oracolo che risolve il problema Y.

Notazione. $X \leq_P Y$.

Nota. Il tempo per scrivere le istanze di Y mandate all'oracolo va conteggiato $\Rightarrow$ le istanze di Y devono avere taglia polinomiale.

Errore da novizi. Confondere $X \leq_P Y$ con $Y \leq_P X$.

Intrattabilità: quiz 1

Supponiamo che $X \leq_{\mathrm{P}} Y$. Cosa si può concludere, tra le seguenti ?

A. Se X può essere risolto in tempo polinomiale, allora anche Y.

B. X è un problema tempo-polinomiale sse Y è tempo-polinomiale.

C. Se X non è risolvibile in tempo polinomiale, allora neanche Y.

D. Se Y non è risolvibile in tempo polinomiale, allora neanche X.

Intrattabilità: quiz 2

Quale delle seguenti riduzioni tempo-polinomiale sono note?

A. FIND-MAX-FLOW $\leq_{\mathrm{P}}$ FIND-MIN-CUT.

B. FIND-MIN-CUT $\leq_{\mathrm{P}}$ FIND-MAX-FLOW.

C. Sia A che B.

D. Né A né B.

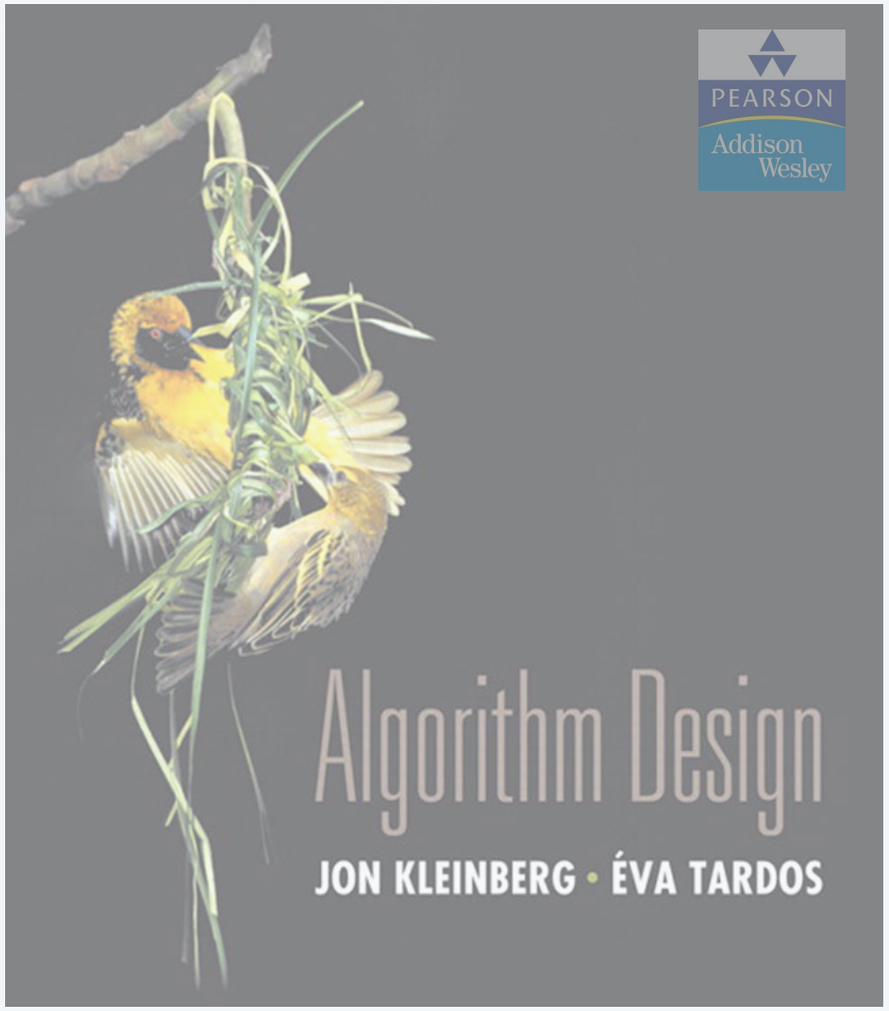
Riduzioni tempo-polinomiali

Progettare algoritmi. Se $X \leq_P Y$ e Y è risolvibile in tempo polinomiale, allora anche X è risolvibile in tempo polinomiale.

Stabilire l'intrattabilità. Se $X \leq_P Y$ e X non è risolvibile in tempo polinomiale, allora neanche Y è risolvibile in tempo polinomiale.

Stabilire l'equivalenza. Se vale sia $X \leq_P Y$ che $Y \leq_P X$, usiamo la notazione $X \equiv_P Y$. In tal caso, X è risolvibile in tempo polinomiale sse lo è Y.

Conclusione. Le riduzioni classificano i problemi in termini di difficoltà relativa.



SECTION 8.1

8. INTRATTABILITÀ I

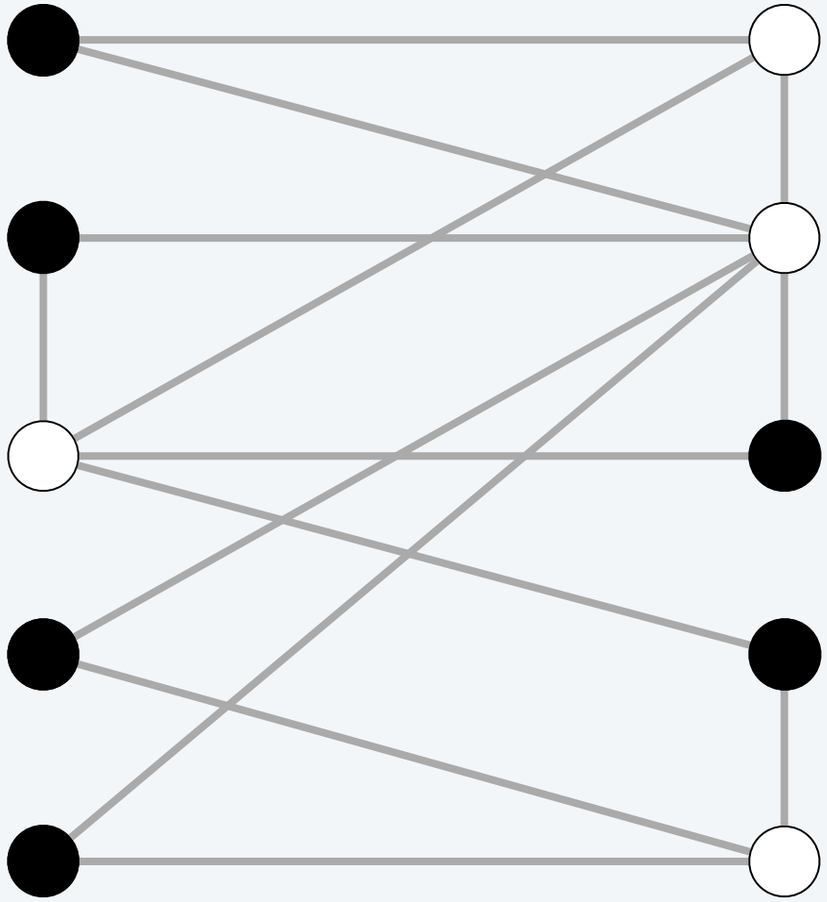
- *poly-time reductions*
- *problemi di copertura e impaccamento*
- *constraint satisfaction problems*
- *sequencing problems*
- *partitioning problems*
- *graph coloring*
- *numerical problems*

Insieme indipendente [Independent set]

INDEPENDENT-SET. Dato un grafo $G = (V, E)$ ed un intero k, esiste un sottoinsieme di k (o più) nodi tali che nessuna coppia è adiacente?

Es. Esiste un insieme indipendente di dimensione ≥ 6 ?

Es. Esiste un insieme indipendente di dimensione ≥ 7 ?



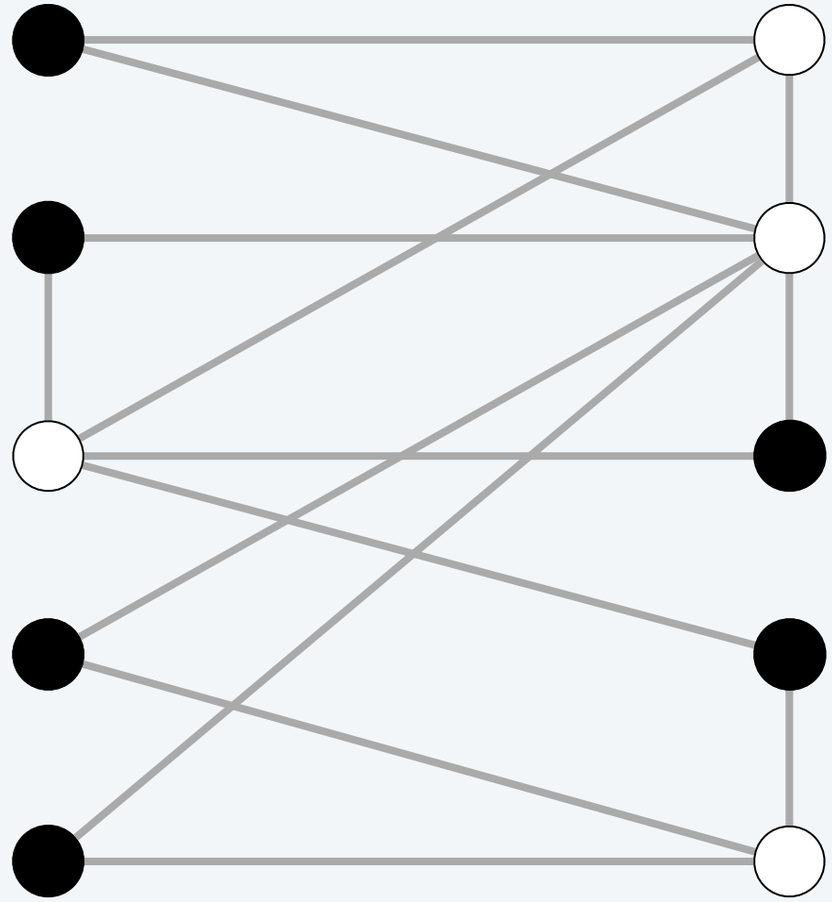
insieme indipendente di dimensione 6

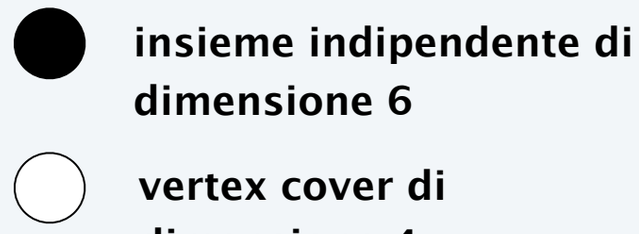
Copertura ai vertici [Vertex cover]

VERTEX-COVER. Dato un grafo $G = (V, E)$ ed un intero k, esiste un sottoinsieme di k (o meno) nodi tali che ogni arco è incidente ad almeno un nodo del sottoinsieme?

Es. Esiste un vertex cover di dimensione ≤ 4 ?

Es. Esiste un vertex cover di dimensione ≤ 3 ?





Intrattabilità: quiz 3

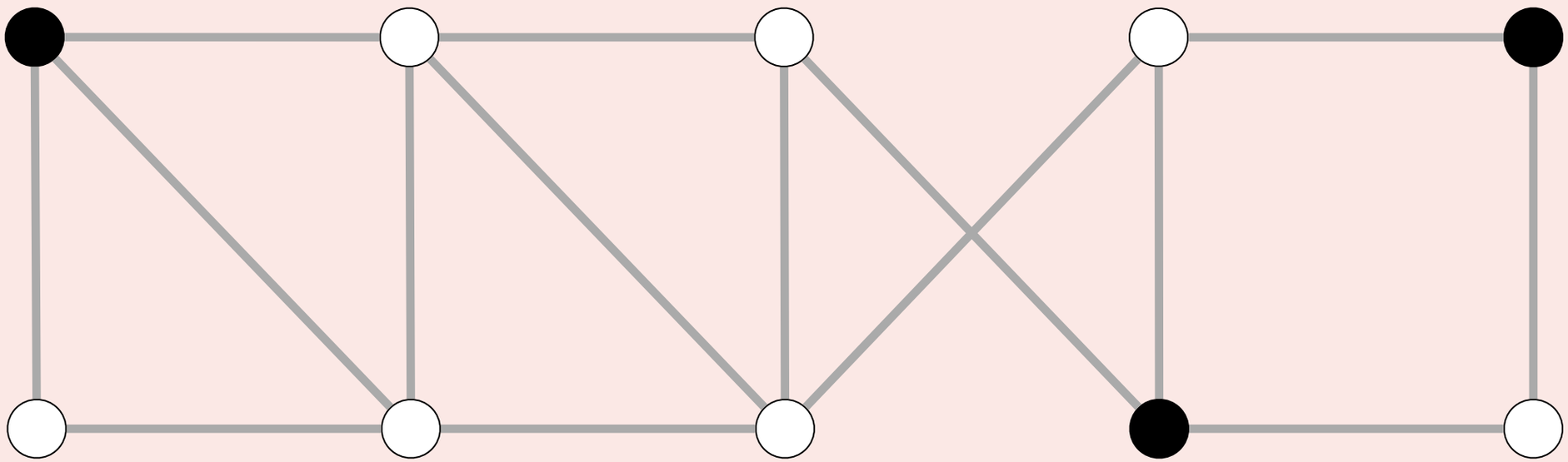
Consideriamo il grafo in figura G. Cosa si può dire?

A. I nodi bianchi sono un vertex cover di dimensione 7.

B. I nodi neri sono un independent set di dimensione 3.

C. Sia A che B.

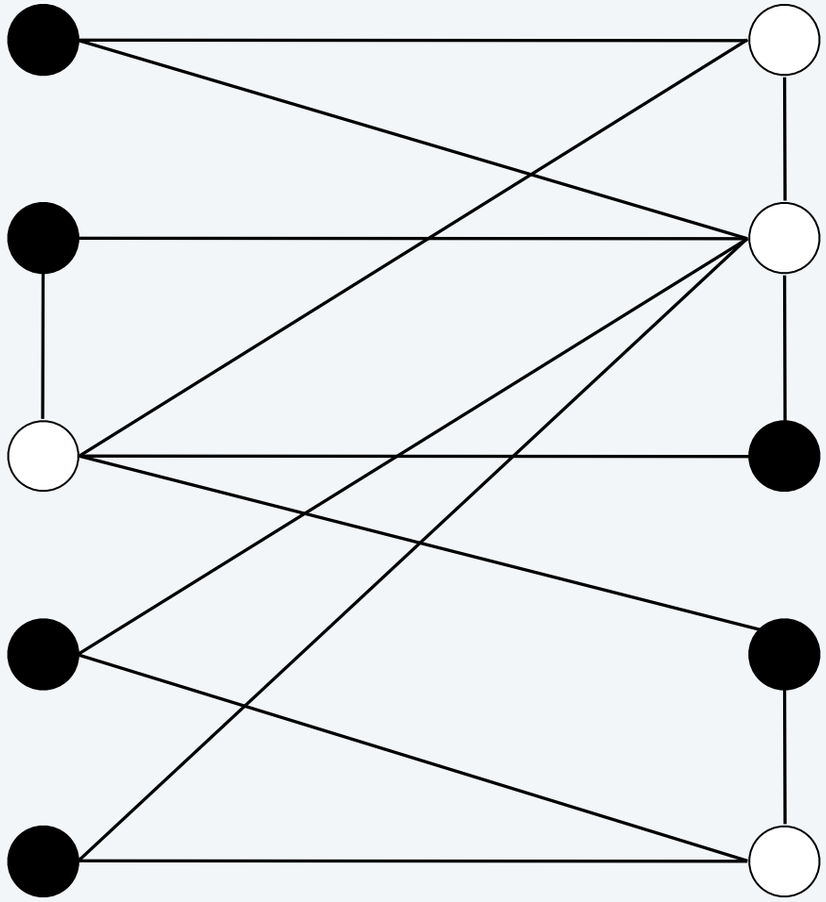
D. Né A né B.

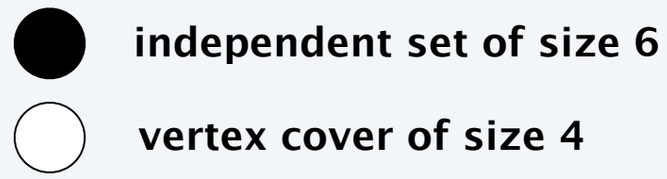


Vertex cover e Independent set sono riducibili l'uno all'altro

Teorema. INDEPENDENT-SET $\equiv_P$ VERTEX-COVER.

Dim. Mostriamo che S è un independent set di dimensione k sse $V - S$ è un vertex cover di dimensione $n - k$.





Vertex cover e Independent set sono riducibili l'uno all'altro

Teorema. INDEPENDENT-SET $\equiv_P$ VERTEX-COVER.

Dim. Mostriamo che S è un independent set di dimensione k sse $V - S$ è un vertex cover di dimensione $n - k$.

$\Rightarrow$

- Sia S un qualunque independent set di dimensione k.
- $V - S$ ha cardinalità $n - k$.
- Consideriamo un arco qualunque $(u, v) \in E$.
- S è indipendente $\Rightarrow$ o $u \notin S$, o $v \notin S$, o entrambe.

  $\Rightarrow$ o $u \in V - S$, o $v \in V - S$, o entrambe.
- Quindi, $V - S$ copre (u, v). ▪

Vertex cover e Independent set sono riducibili l'uno all'altro

Teorema. INDEPENDENT-SET $\equiv_P$ VERTEX-COVER.

Dim. Mostriamo che S è un independent set di dimensione k sse $V - S$ è un vertex cover di dimensione $n - k$.

$\Leftarrow$

- Sia $V - S$ un qualunque vertex cover di dimensione $n - k$.
- S ha cardinalità k.
- Consideriamo un qualunque arco $(u, v) \in E$.
- $V - S$ è un vertex cover $\Rightarrow$ o $u \in V - S$, o $v \in V - S$, o entrambe.
  $\Rightarrow$ o $u \notin S$, o $v \notin S$, o entrambe.
- Quindi, S è un independent set. ▪

Copertura di insieme [Set cover]

SET-COVER. Dato un insieme U di elementi, una collezione S di sottoinsiemi di U, ed un intero k, esistono $\leq k$ di quei sottoinsiemi con unione U?

Esempio di applicazione.

- m programmi software disponibili.
- Insieme U di n funzionalità che vorremmo fornire al nostro sistema.
- Il programma i-esimo fornisce il sottoinsieme di funzionalità $S_i \subseteq U$.
- Scopo: ottenere tutte le n funzionalità utilizzando meno programmi possibili.

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 3, 7 \}$ $S_b = \{ 2, 4 \}$

$S_c = \{ 3, 4, 5, 6 \}$ $S_d = \{ 5 \}$

$S_e = \{ 1 \}$ $S_f = \{ 1, 2, 6, 7 \}$

$k = 2$

un'istanza di set cover

Intrattabilità: quiz 4

Dato l'universo U = { 1, 2, 3, 4, 5, 6, 7 } ed i seguenti insiemi, qual è la dimensione minima di un set cover?

A. 1

B. 2

C. 3

D. Nessuna delle precedenti.

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 1, 4, 6 \}$ $S_b = \{ 1, 6, 7 \}$

$S_c = \{ 1, 2, 3, 6 \}$ $S_d = \{ 1, 3, 5, 7 \}$

$S_e = \{ 2, 6, 7 \}$ $S_f = \{ 3, 4, 5 \}$

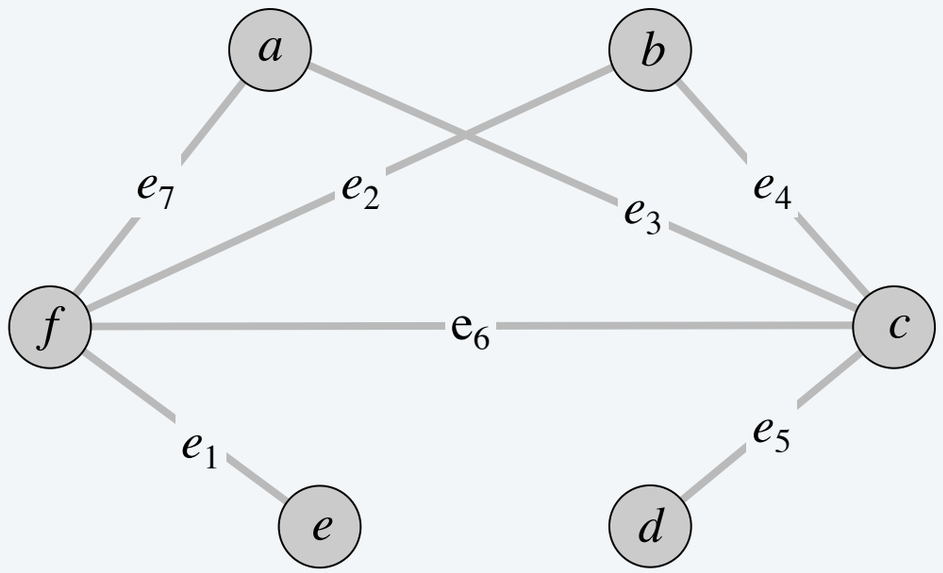
Vertex cover è riducibile al Set cover

Teorema. VERTEX-COVER $\leq_P$ SET-COVER.

Dim. Data un'istanza VERTEX-COVER $G = (V, E)$ e k, costruiamo un'istanza di SET-COVER (U, S, k) che ha un set cover di dimensione k sse G ha un vertex cover di dimensione k.

Costruzione.

- Universo $U = E$.
- Includi un sottoinsieme per ogni nodo $v \in V$: $S_v = \{e \in E : e \text{ incident to } v\}$.



istanza di vertex cover
(k = 2)

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 3, 7 \}$ $S_b = \{ 2, 4 \}$

$S_c = \{ 3, 4, 5, 6 \}$ $S_d = \{ 5 \}$

$S_e = \{ 1 \}$ $S_f = \{ 1, 2, 6, 7 \}$

istanza di set cover
(k = 2)

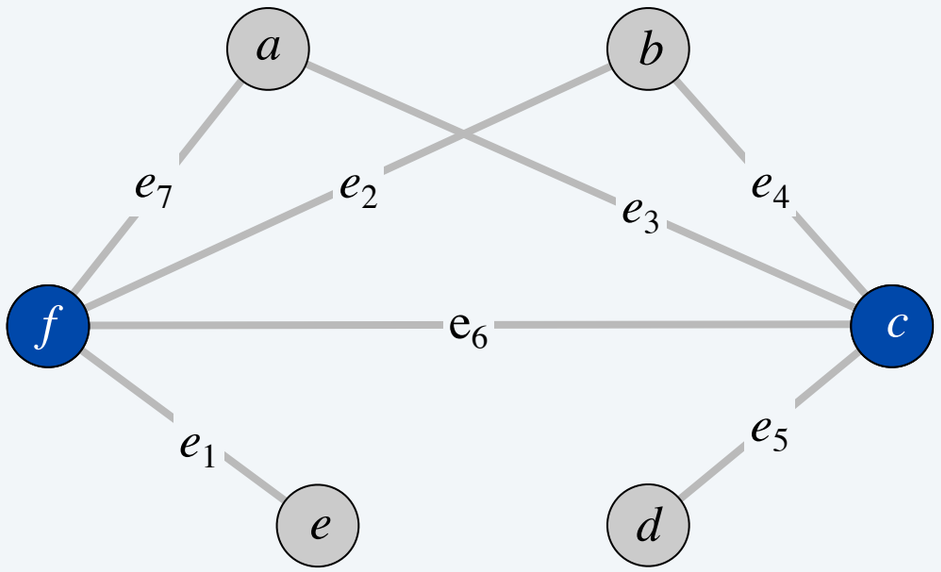
Vertex cover è riducibile al Set cover

Lemma. $G = (V, E)$ contiene un vertex cover di dimensione k sse (U, S, k) contiene un set cover di dimensione k.

Dim. $\Rightarrow$ Sia $X \subseteq V$ un vertex cover di taglia k in G.

- Allora $Y = \{ S_v : v \in X \}$ è un set cover di taglia k. ▪

le istanze "yes" di VERTEX-COVER sono risolte correttamente



istanza di vertex cover (k = 2)

$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 3, 7 \}$ $S_b = \{ 2, 4 \}$

$S_c = \{ 3, 4, 5, 6 \}$ $S_d = \{ 5 \}$

$S_e = \{ 1 \}$ $S_f = \{ 1, 2, 6, 7 \}$

istanza di set cover (k = 2)

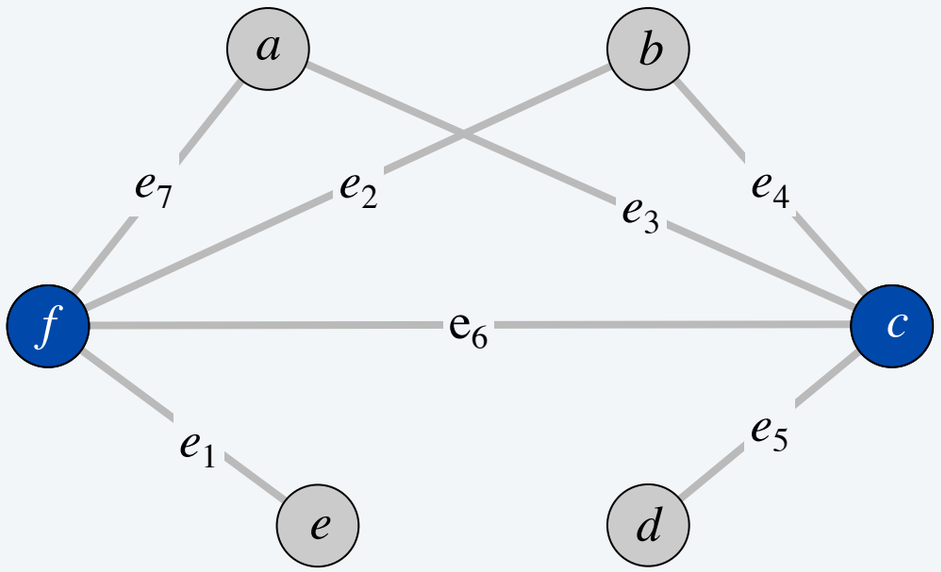
Vertex cover è riducibile al Set cover

Lemma. $G = (V, E)$ contiene un vertex cover di dimensione k sse (U, S, k) contiene un set cover di dimensione k.

Dim. $\Leftarrow$ Sia $Y \subseteq S$ un set cover di taglia k in (U, S, k).

- Allora $X = \{ v : S_v \in Y \}$ è un vertex cover di taglia k in G. ▪

istanze "no" di VERTEX-COVER sono risolte correttamente



istanza di vertex cover (k = 2)

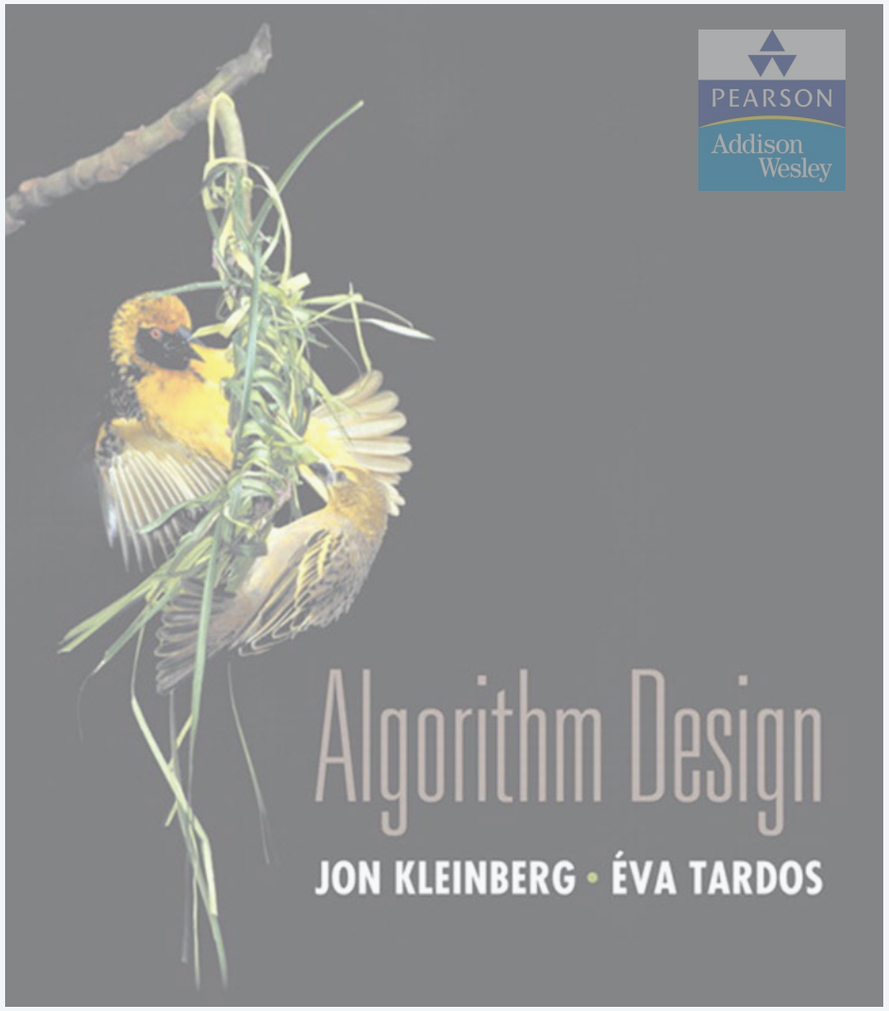
$U = \{ 1, 2, 3, 4, 5, 6, 7 \}$

$S_a = \{ 3, 7 \}$ $S_b = \{ 2, 4 \}$

$S_c = \{ 3, 4, 5, 6 \}$ $S_d = \{ 5 \}$

$S_e = \{ 1 \}$ $S_f = \{ 1, 2, 6, 7 \}$

istanza di set cover (k = 2)



SECTION 8.2

8. INTRATTABILITÀ I

- *poly-time reductions*
- *packing and covering problems*
- *problemi di soddisfacimento di vincoli*
- *sequencing problems*
- *partitioning problems*
- *graph coloring*
- *numerical problems*

Soddisfacibilità [Satisfiability]

Letterale. Una variabile booleana o la sua negata. x_i or $\overline{x_i}$

Clausola. Una disgiunzione di letterali. $C_j = x_1 \vee \overline{x_2} \vee x_3$

Forma normale congiuntiva (CNF). Una formula proposizionale Φ che è una congiunzione di clausole. $\Phi = C_1 \wedge C_2 \wedge C_3 \wedge C_4$

SAT. Data una formula CNF Φ, ammette un'assegnazione che la soddisfa?

3-SAT. SAT in cui ogni clausola contiene esattamente 3 letterali (ed ogni letterale corrisponde ad una variabile distinta).

$$\Phi = \left(\overline{x_1} \vee x_2 \vee x_3 \right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3 \right) \wedge \left(\overline{x_1} \vee x_2 \vee x_4 \right)$$

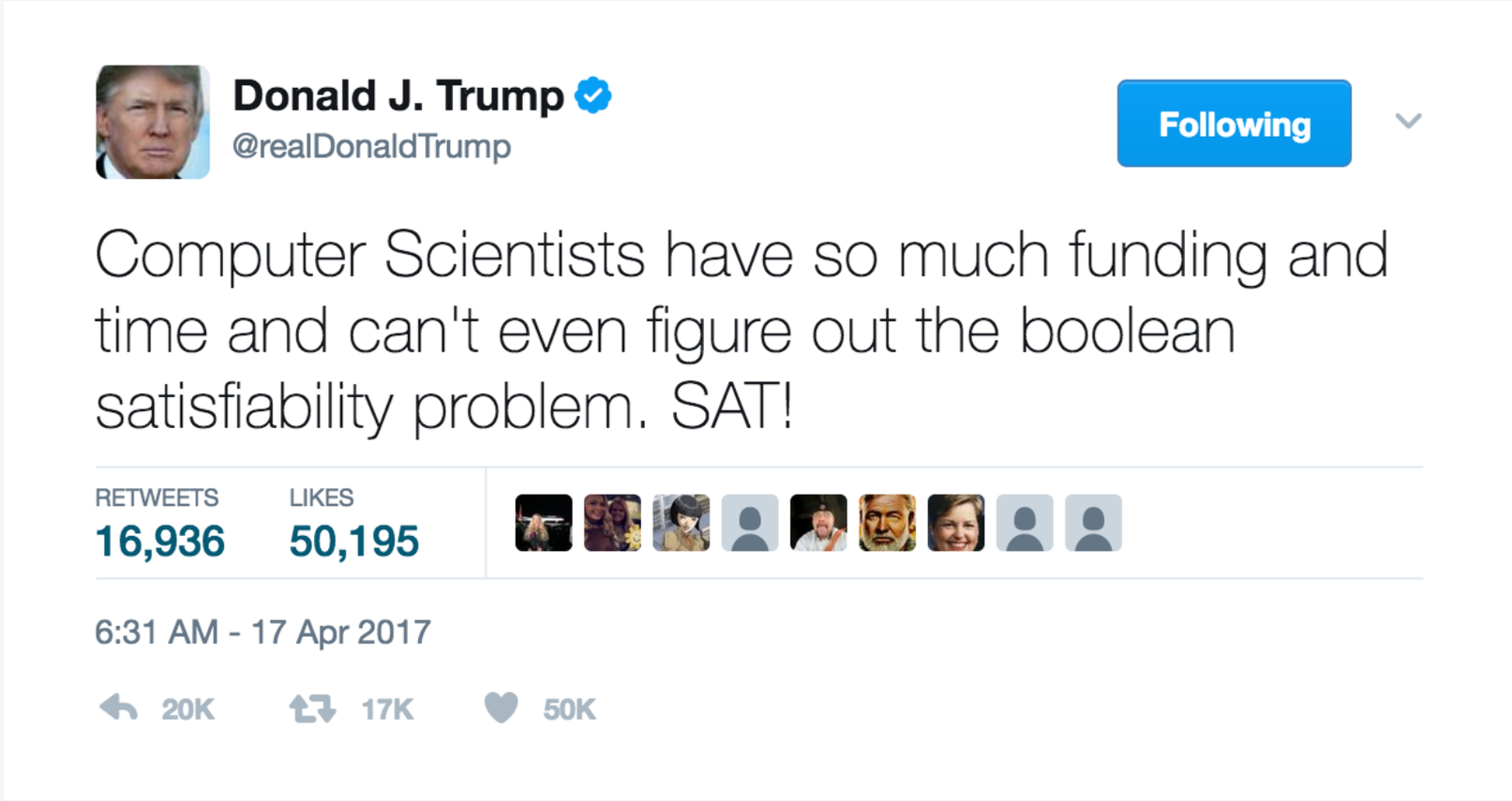
istanza yes: x_1 = true, x_2 = true, x_3 = false, x_4 = false

Applicazione chiave. Automazione di progetti elettronici (EDA).

Satisfiability è difficile

Ipotesi scientifica. Non esiste un algoritmo polinomiale per 3-SAT.

P vs. NP. Questa ipotesi è equivalente alla congettura **P ≠ NP**.



https://www.facebook.com/pg/npcompleteteens

3-satisfiability è riducibile a Independent set

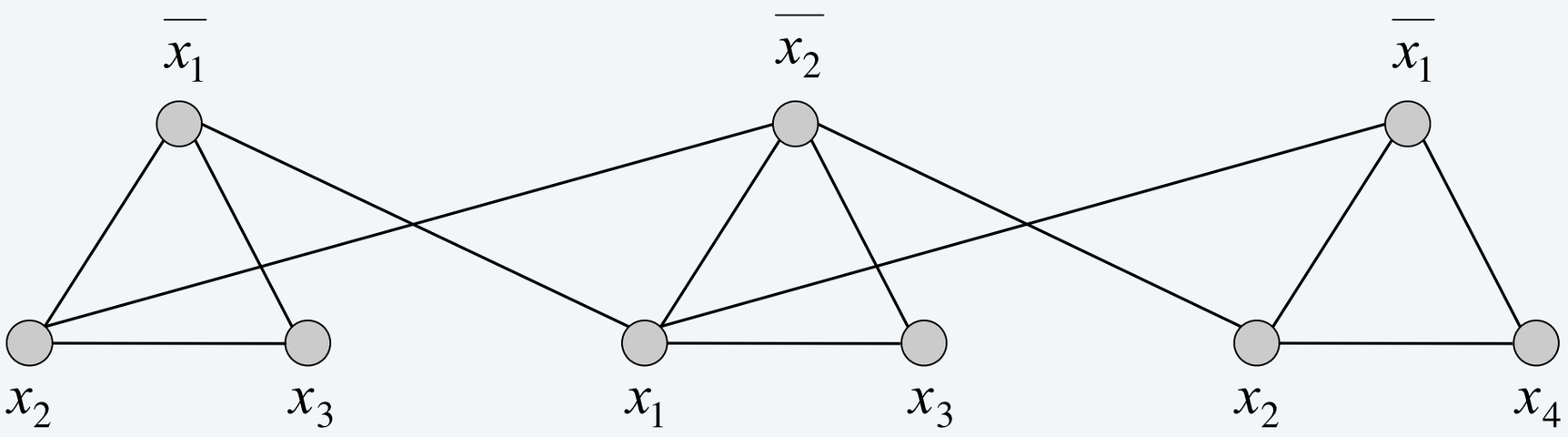
Teorema. 3-SAT $\leq_P$ INDEPENDENT-SET.

Dim. Data un'istanza Φ di 3-SAT, costruiamo un'istanza (G, k) di INDEPENDENT-SET che ha un independent set di taglia $k = |\Phi|$ sse Φ è soddisfacibile.

Costruzione.

- G contiene 3 nodi per ogni clausola, uno per ciascun letterale.
- Connetti 3 letterali di ogni clausola in un triangolo.
- Connetti ogni letterale a ciascuna delle sue negazioni.

G



k = 3

$$\Phi = \left(\overline{x_1} \vee x_2 \vee x_3 \right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3 \right) \wedge \left(\overline{x_1} \vee x_2 \vee x_4 \right)$$

3-satisfiability è riducibile a Independent set

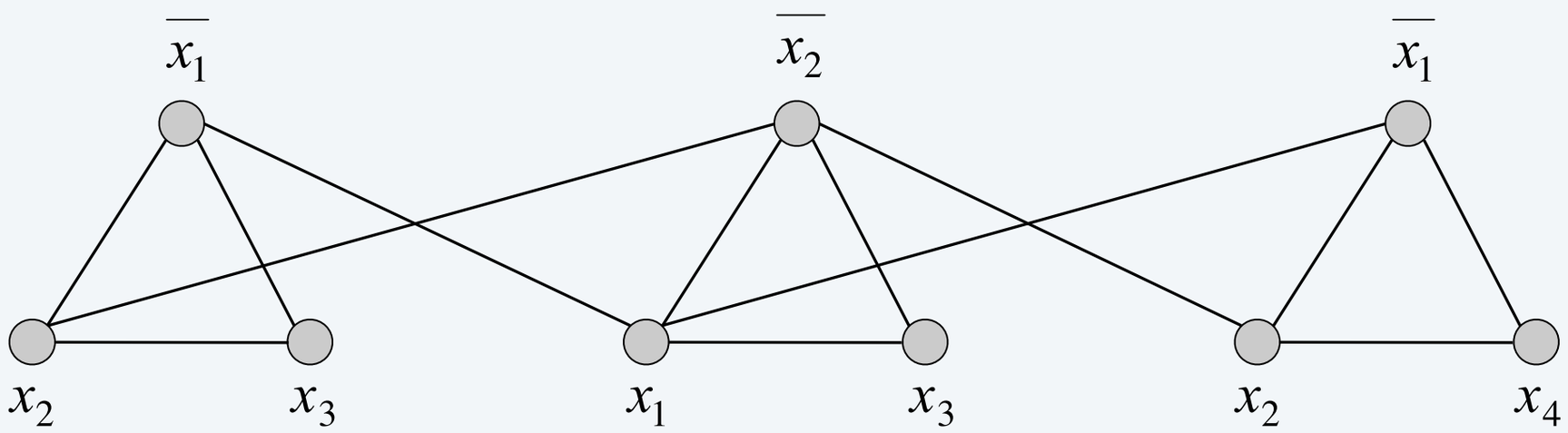
Lemma. Φ è soddisfacibile sse G contiene un independent set di taglia $k = |\Phi|$.

Dim. $\Rightarrow$ Considera un assegnazione che soddisfa Φ.

- Scegli un letterale vero da ogni clausola/triangolo.
- Ciò forma un independent set di taglia $k = |\Phi|$. ▪

le istanze "yes" di 3-SAT sono risolte correttamente

G



k = 3

$$\Phi = \left(\overline{x_1} \vee x_2 \vee x_3 \right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3 \right) \wedge \left(\overline{x_1} \vee x_2 \vee x_4 \right)$$

3-satisfiability è riducibile a Independent set

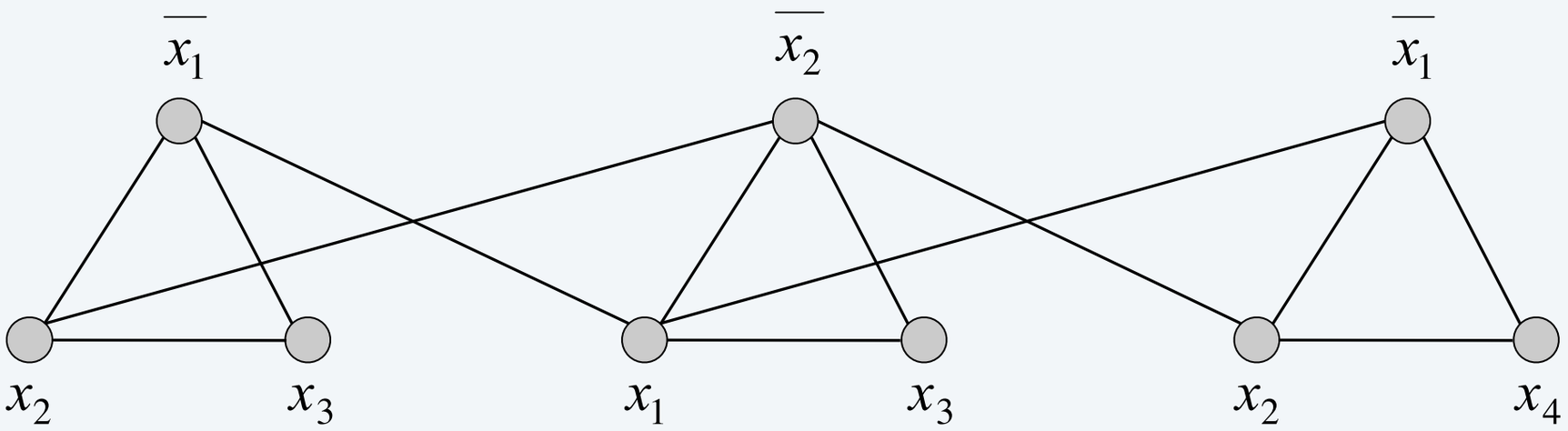
Lemma. Φ è soddisfacibile sse G contiene un independent set di taglia $k = |\Phi|$.

Dim. $\Leftarrow$ Sia S un independent set di taglia k.

- S deve contenere esattamente un nodo per ogni triangolo.
- Poni questi letterali a *true* (e i letterali rimanenti in modo coerente).
- Tutte le clausole di Φ sono soddisfatte. ▪

le istanze "no" di 3-SAT sono risolte correttamente

G



k = 3

$$\Phi = \left(\overline{x_1} \vee x_2 \vee x_3 \right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3 \right) \wedge \left(\overline{x_1} \vee x_2 \vee x_4 \right)$$

Riepilogo

Strategie fondamentali di riduzione

- Semplice equivalenza: INDEPENDENT-SET $\equiv_P$ VERTEX-COVER.
- Da caso particolare a caso generale: VERTEX-COVER $\leq_P$ SET-COVER.
- Codifica attraverso "gadget": 3-SAT $\leq_P$ INDEPENDENT-SET.

Transitività. Se $X \leq_P Y$ e $Y \leq_P Z$, allora $X \leq_P Z$.

Pf (intuizione). Componiamo i due algoritmi.

Es. 3-SAT $\leq_P$ INDEPENDENT-SET $\leq_P$ VERTEX-COVER $\leq_P$ SET-COVER.

Problemi di decisione, ricerca, ed ottimizzazione

Problema di decisione. Esiste un vertex cover di taglia $\leq k$?

Problema di ricerca. Trova un vertex cover di taglia $\leq k$.

Problema di ottimizzazione. Trova un vertex cover di taglia minima.

Scopo. Mostriamo che i tre problemi sono riducibili uno all'altro in tempo polinomiale.

PROBLEMI DI RICERCA VS. PROBLEMI DI DECISIONE

VERTEX-COVER. Esiste un vertex cover di taglia $\leq k$?

FIND-VERTEX-COVER. Trova un vertex cover di taglia $\leq k$.

Teorema. VERTEX-COVER $\equiv_P$ FIND-VERTEX-COVER.

Dim. $\leq_P$ Il problema di decisione è un caso particolare del problema di ricerca. ▪

Dim. $\geq_P$

Per trovare un vertex cover di taglia $\leq k$:

- Determina se esiste un vertex cover di taglia $\leq k$.
- Trova un nodo v tale che $G - \{ v \}$ ha un vertex cover di taglia $\leq k - 1$.
  (ogni nodo di un vertex cover di taglia $\leq k$ ha questa proprietà)
- Includi v nel vertex cover.
- Trova ricorsivamente un vertex cover di taglia $\leq k - 1$ in $G - \{ v \}$. ▪

cancella v e tutti gli archi incidenti

PROBLEMI DI OTTIMIZZAZIONE VS. PROBLEMI DI RICERCA

FIND-VERTEX-COVER. Trova un vertex cover di taglia $\leq k$.

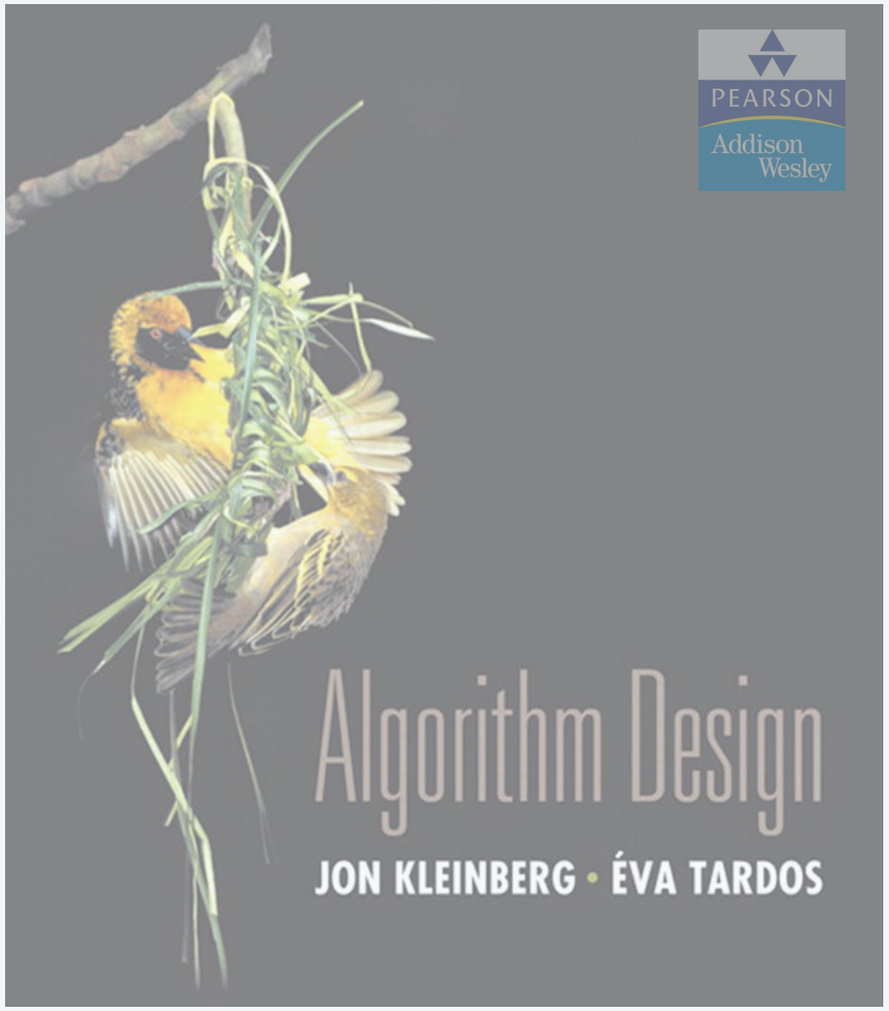
FIND-MIN-VERTEX-COVER. Trova un vertex cover di taglia minima.

Teorema. FIND-VERTEX-COVER $\equiv_P$ FIND-MIN-VERTEX-COVER.

Dim. $\leq_P$ Il problema di ricerca è un caso particolare del problema di ottimizzazione. ▪

Dim. $\geq_P$ Per trovare un vertex cover di taglia minima:

- Ricerca binaria (o sequenziale) per la taglia k^* del vertex cover minimo.
- Risolvi il problema di ricerca per il k^* dato. ▪



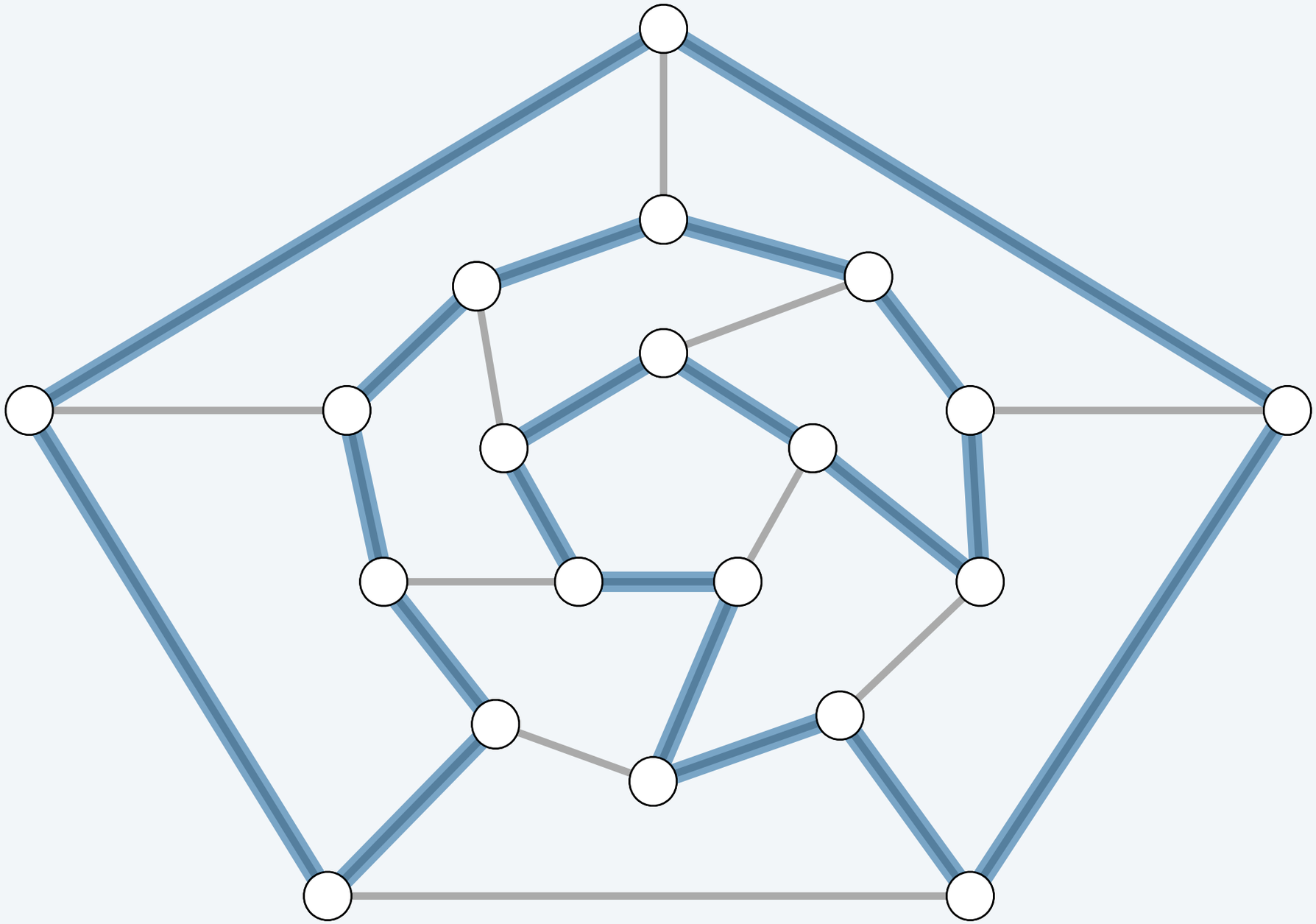
SECTION 8.5

8. INTRATTABILITÀ I

- *poly-time reductions*
- *packing and covering problems*
- *constraint satisfaction problems*
- *problemi di sequenziamento*
- *partitioning problems*
- *graph coloring*
- *numerical problems*

Ciclo Hamiltoniano [Hamilton cycle]

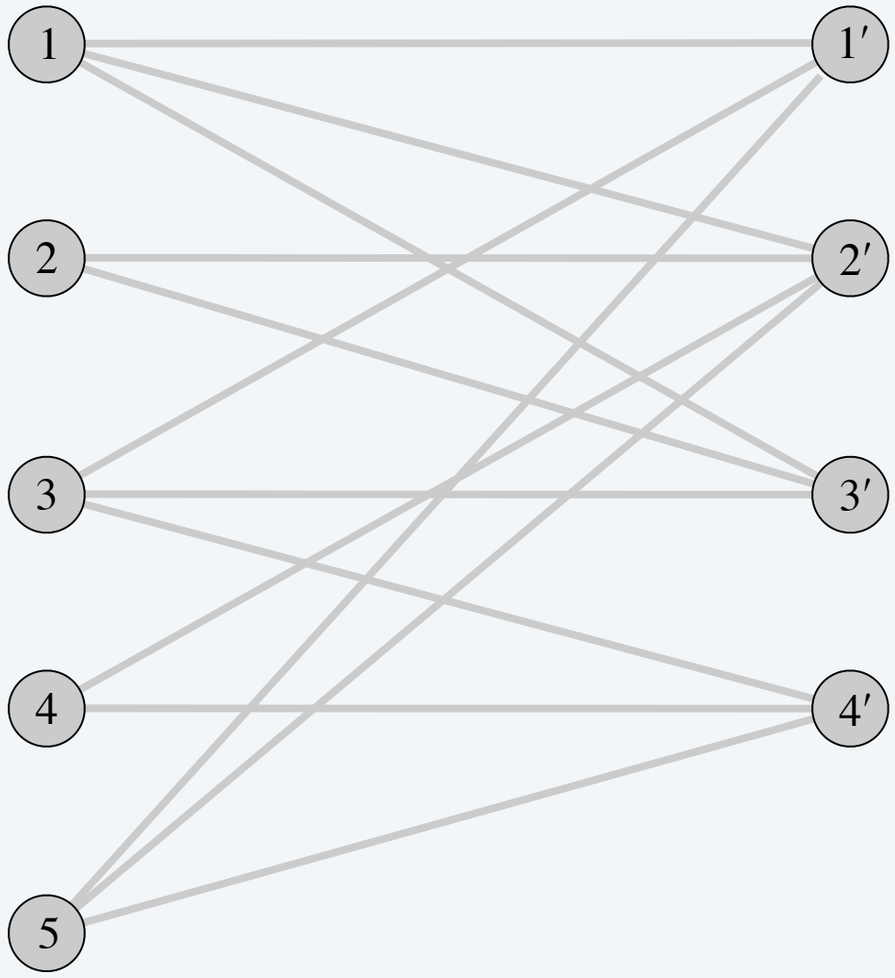
HAMILTON-CYCLE. Dato un grafo non orientato $G = (V, E)$, esiste un ciclo Γ che visiti ogni nodo esattamente una volta?



yes

Ciclo Hamiltoniano

HAMILTON-CYCLE. Dato un grafo non orientato $G = (V, E)$, esiste un ciclo Γ che visiti ogni nodo esattamente una volta?



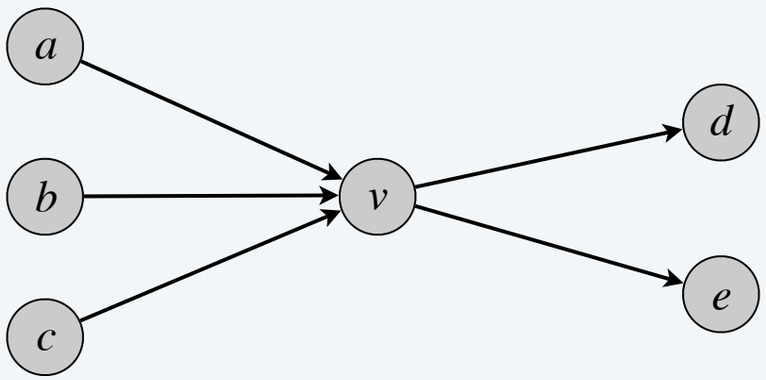
no

Ciclo Hamiltoniano orientato è riducibile a Ciclo Hamiltoniano

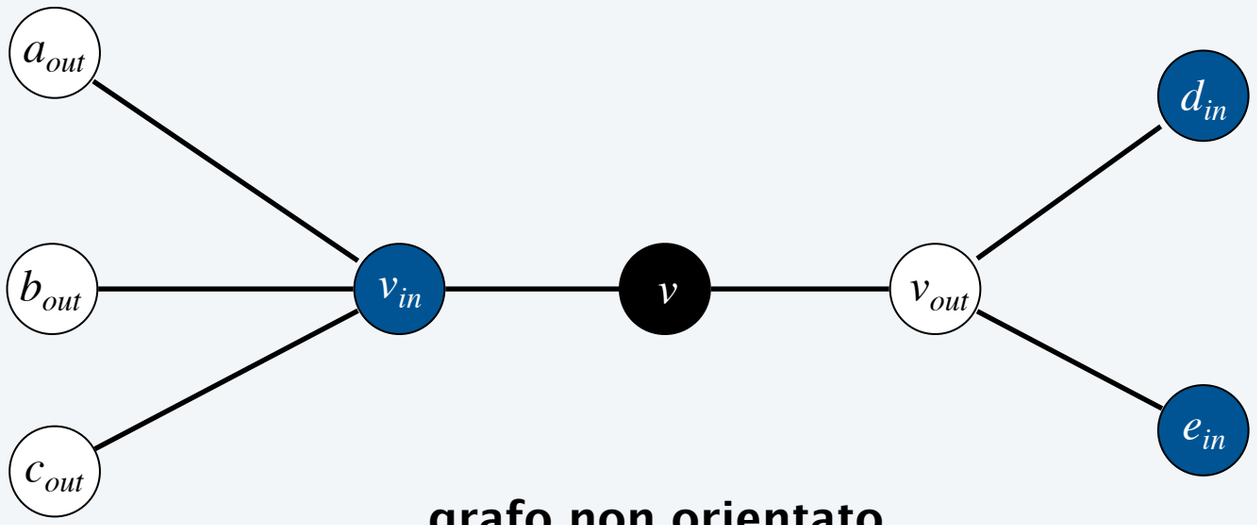
DIRECTED-HAMILTON-CYCLE. Dato un digrafo $G = (V, E)$, esiste un ciclo orientato Γ che visiti ogni nodo esattamente una volta?

Teorema. DIRECTED-HAMILTON-CYCLE $\leq_P$ HAMILTON-CYCLE.

Dim. Dato un digrafo $G = (V, E)$, costruisci un grafo G' con $3n$ nodi.



digrafo G



grafo non orientato G′

Ciclo Hamiltoniano orientato è riducibile a Ciclo Hamiltoniano

Lemma. G ha un ciclo hamiltoniano orientato sse G' ha un ciclo hamiltoniano.

Dim. $\Rightarrow$

- Supponi che G abbia un ciclo hamiltoniano orientato Γ.
- Allora G' ha un ciclo hamiltoniano non orientato (stessa sequenza). ▪

Dim. $\Leftarrow$

- Supponi che G' abbia un ciclo hamiltoniano non orientato Γ'.
- Γ' deve visitare i nodi in G' in una delle seguenti due sequenze:

  $\ldots, nero, bianco, blu, nero, bianco, blu, nero, bianco, blu, \ldots$

  $\ldots, nero, blu, bianco, nero, blu, bianco, nero, blu, bianco, \ldots$
- I nodi neri di Γ' o formano un ciclo hamiltoniano orientato Γ in G, o formano il rovescio di un ciclo hamiltoniano orientato. ▪

3-satisfiability è riducibile a Ciclo hamiltoniano orientato

Teorema. 3-SAT $\leq_P$ DIRECTED-HAMILTON-CYCLE.

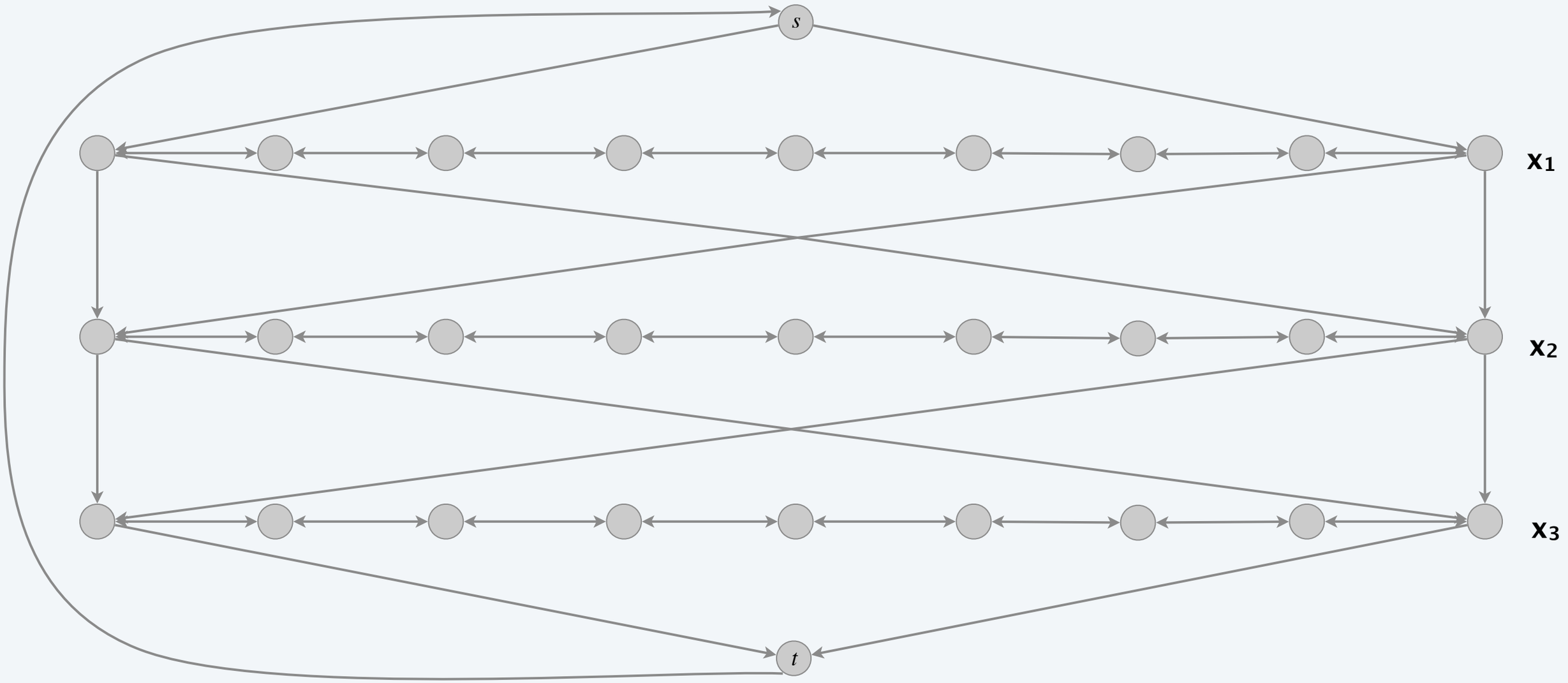
Dim. Data un'istanza Φ di 3-SAT, costruiamo un'istanza G di DIRECTED-HAMILTON-CYCLE che ha un ciclo hamiltoniano sse Φ è soddisfacibile.

Anteprima della costruzione. Sia n il numero di variabili di Φ.
Costruiremo un grafo G che ha 2^n cicli hamiltoniani, con ogni ciclo in corrispondenza con una delle 2^n possibili assegnazioni di verità.

3-satisfiability è riducibile a Ciclo hamiltoniano orientato

Costruzione. Data: un'istanza 3-SAT Φ con n variabili x_i e k clausole.

- Vogliamo costruire G con 2^n cicli hamiltoniani.
- Intuizione: attraverso cammino i da sx a dx $\Leftrightarrow$ pongo variabile $x_i = true$.



Intrattabilità: quiz 5

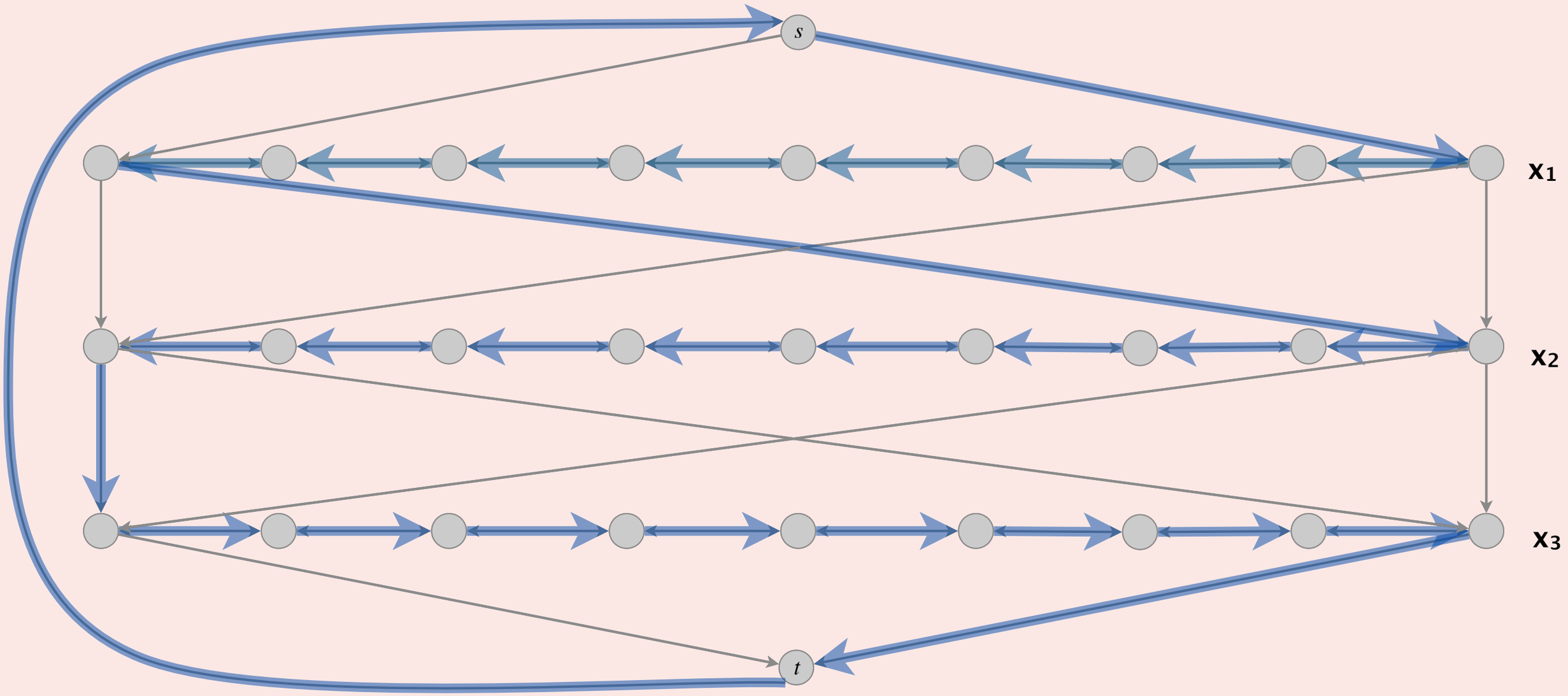
Quale assegnazione di verità corrisponde al ciclo in figura?

A. $x_1 = true, x_2 = true, x_3 = true$

B. $x_1 = true, x_2 = true, x_3 = false$

C. $x_1 = false, x_2 = false, x_3 = true$

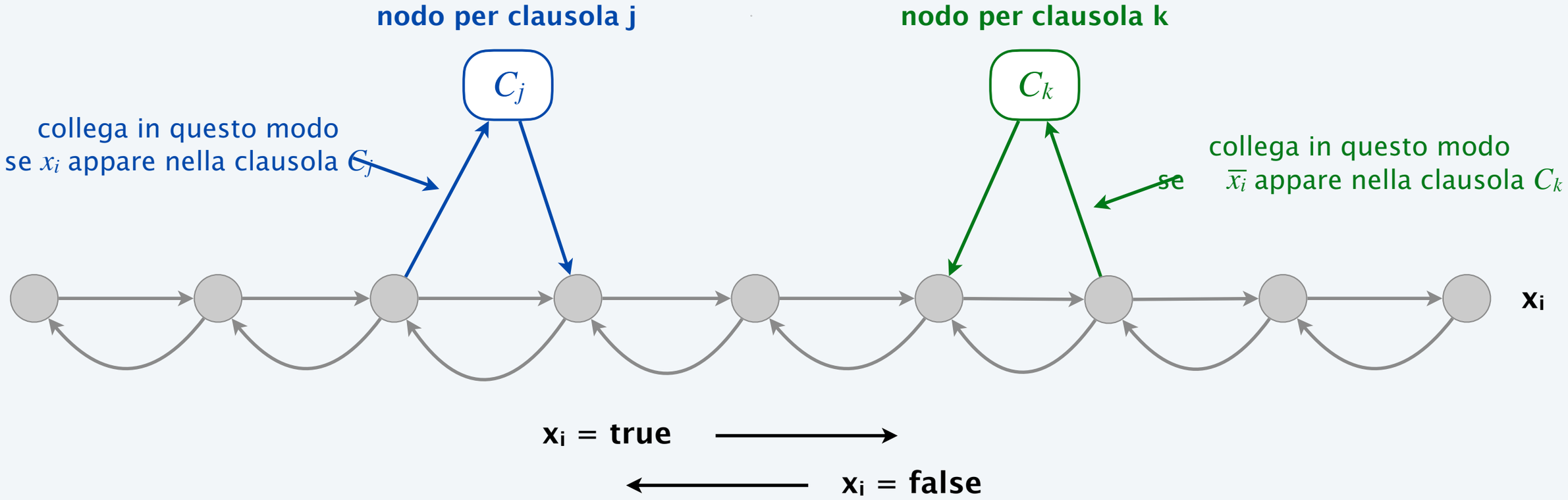
D. $x_1 = false, x_2 = false, x_3 = false$



3-satisfiability è riducibile a Ciclo hamiltoniano orientato

Costruzione. Data un'istanza 3-SAT Φ con n variabili x_i e k clausole.

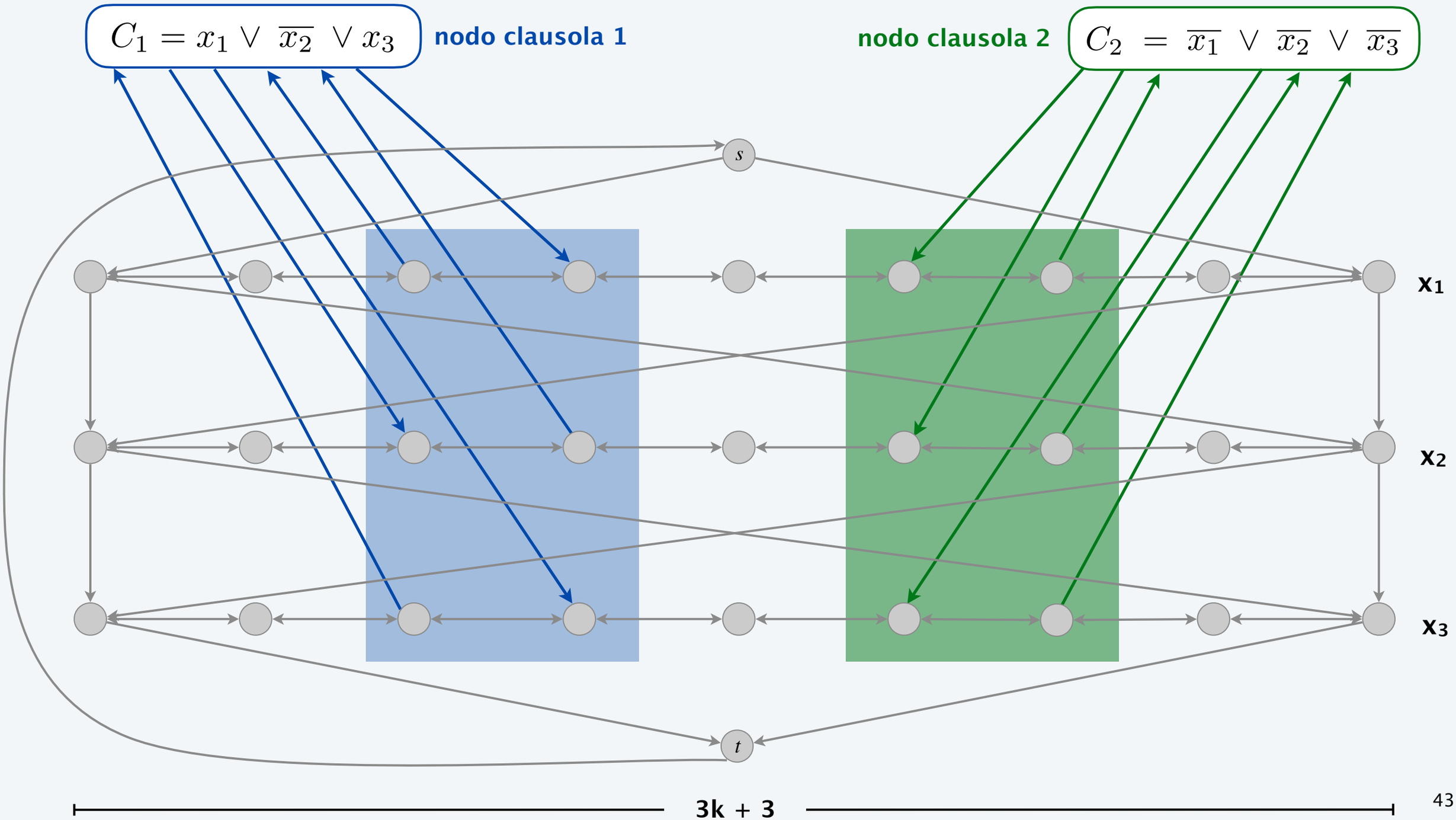
- Per ogni clausola: aggiungi un nodo e 2 archi per letterale.



3-satisfiability è riducibile a Ciclo hamiltoniano orientato

Costruzione. Data un'istanza 3-SAT Φ con n variabili x_i e k clausole.

- Per ogni clausola: aggiungi un nodo e 2 archi per letterale.



3-satisfiability è riducibile a Ciclo hamiltoniano orientato

Lemma. Φ è soddisfacibile sse G ha un ciclo hamiltoniano.

Dim. $\Rightarrow$

- Supponi che l'istanza di 3-SAT Φ sia soddisfatta dall'assegnazione x^*.
- Definisci il ciclo hamiltoniano Γ in G come segue:
  - Se $x_i^* = true$, attraversa la riga i da sinistra a destra
  - Se $x_i^* = false$, attraversa la riga i da destra a sinistra
  - Per ogni clausola C_j, ci sarà almeno una riga i in cui andiamo nel verso "corretto" inserendo la clausola C_j nel ciclo (e inseriamo C_j esattamente una volta) ▪

3-satisfiability è riducibile a Ciclo hamiltoniano orientato

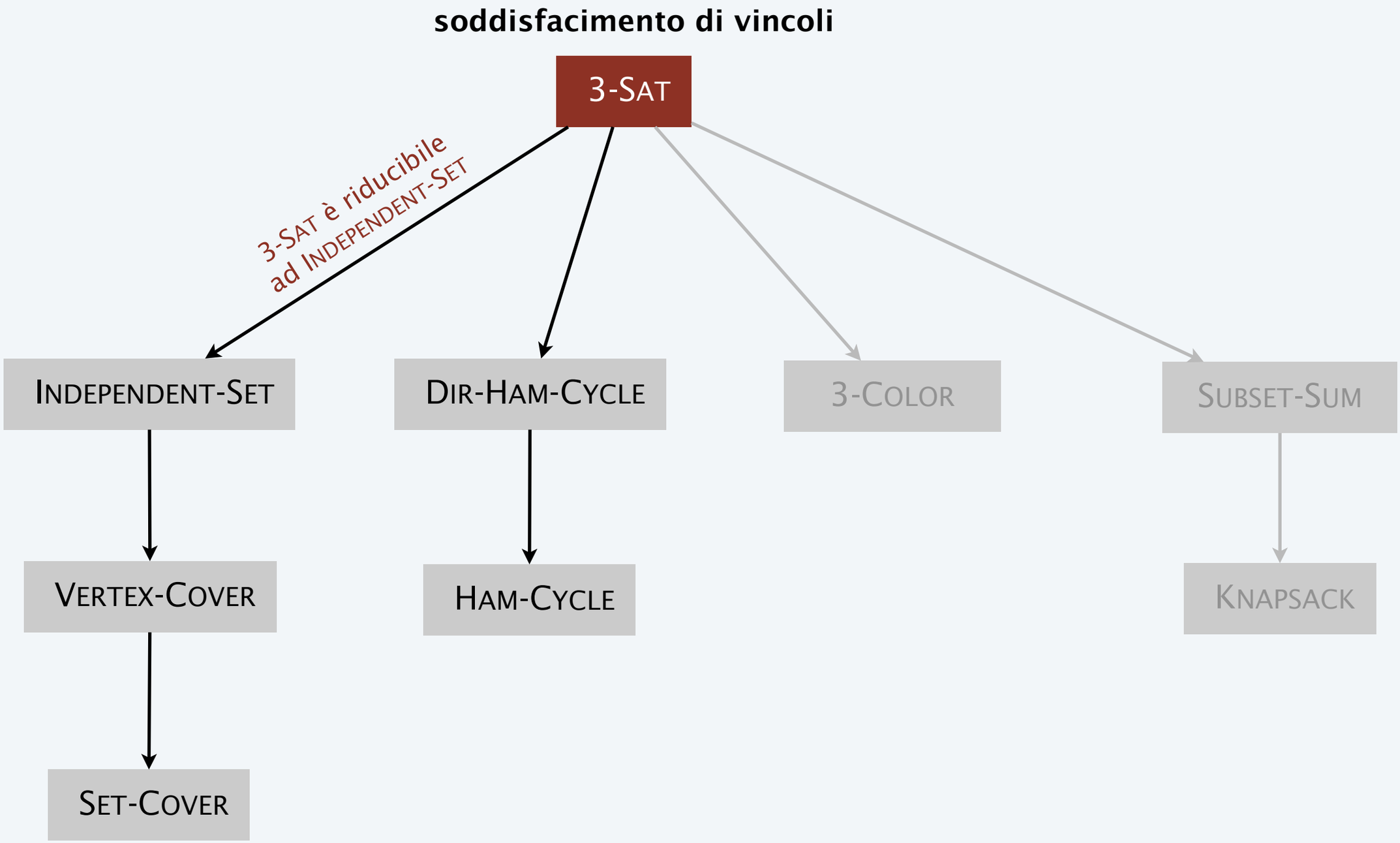
Lemma. Φ è soddisfacibile sse G ha un ciclo hamiltoniano.

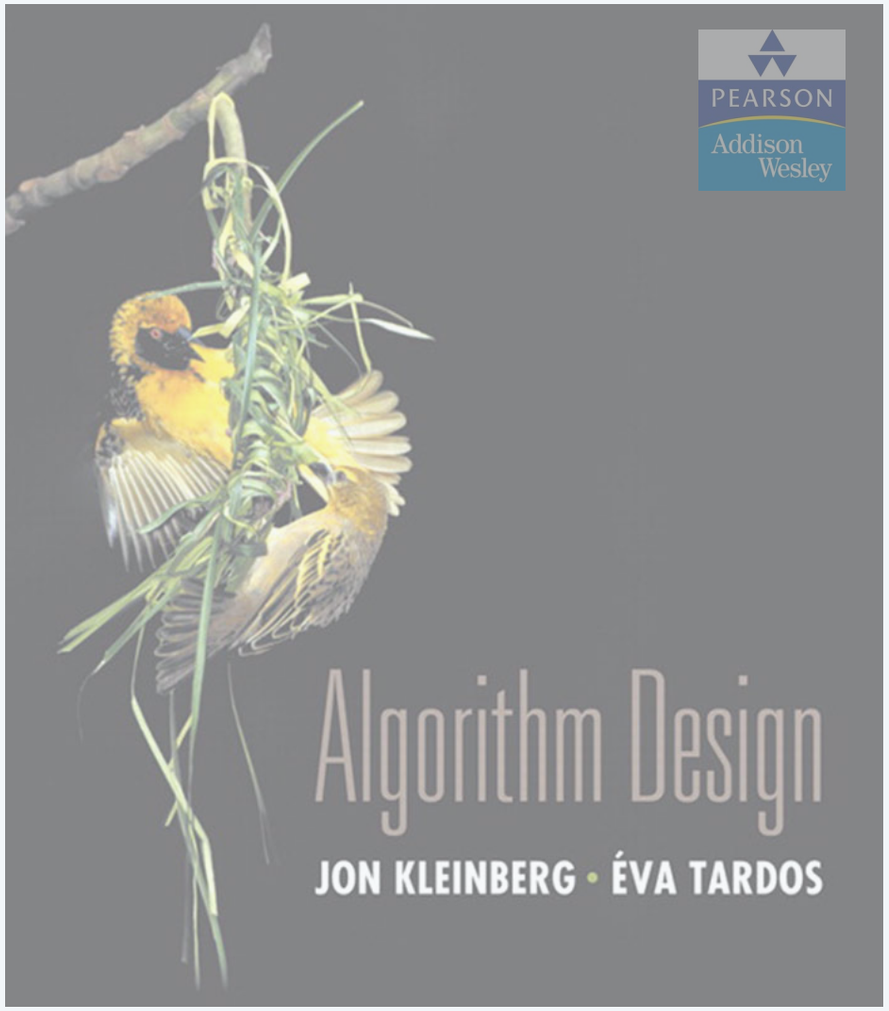
Dim. $\Leftarrow$

- Supponi che G abbia un ciclo hamiltoniano Γ.
- Se Γ entra nel nodo clausola C_j, deve uscirne dall'arco gemello.
  - i nodi subito prima e subito dopo C_j sono collegati da un arco $e \in E$
  - rimuovendo C_j dal ciclo, e rimpiazzandolo con l'arco e si ottiene un ciclo hamiltoniano su $G - \{ C_j \}$
- Continuando in questo modo, si rimane con un ciclo hamiltoniano Γ' su $G - \{ C_1, C_2, \ldots, C_k \}$.
- Poni $x_i^* = true$ se Γ' attraversa la riga i da sx a dx; altrimenti, poni $x_i^* = false$.
- L'attraversamento è nel verso “corretto”, ed ogni clausola è soddisfatta.

■

Riduzioni tempo-polinomiali





SECTION 8.6

8. INTRATTABILITÀ I

- *poly-time reductions*
- *packing and covering problems*
- *constraint satisfaction problems*
- *sequencing problems*
- *problemi di partizionamento*
- *graph coloring*
- *numerical problems*

Abbinamento tridimensionale [3D-Matching]

3D-MATCHING. Dati n docenti, n corsi, n orari, ed una lista dei possibili corsi ed orari che ogni docente è disposto ad insegnare, è possibile trovare un'assegnazione in cui tutti i corsi sono insegnati in orari diversi?

docente	corso	orario
Wayne	COS 226	TTh 11–12:20
Wayne	COS 423	MW 11–12:20
Wayne	COS 423	TTh 11–12:20
Tardos	COS 423	TTh 3–4:20
Tardos	COS 523	TTh 3–4:20
Kleinberg	COS 226	TTh 3–4:20
Kleinberg	COS 226	MW 11–12:20
Kleinberg	COS 423	MW 11–12:20

Abbinamento tridimensionale

3D-MATCHING. Dati 3 insiemi disgiunti X, Y, e Z, ognuno di taglia n ed un insieme $T \subseteq X \times Y \times Z$ di triple, esiste un insieme di n triple di T tale che ogni elemento di $X \cup Y \cup Z$ appartiene ad esattamente una di quelle triple?

$X = \{ x_1, x_2, x_3 \}$, $Y = \{ y_1, y_2, y_3 \}$, $Z = \{ z_1, z_2, z_3 \}$

$T_1 = \{ x_1, y_1, z_2 \}$, $T_2 = \{ x_1, y_2, z_1 \}$, $T_3 = \{ x_1, y_2, z_2 \}$

$T_4 = \{ x_2, y_2, z_3 \}$, $T_5 = \{ x_2, y_3, z_3 \}$,

$T_7 = \{ x_3, y_1, z_3 \}$, $T_8 = \{ x_3, y_1, z_1 \}$, $T_9 = \{ x_3, y_2, z_1 \}$

un'istanza di 3d-matching (con n = 3)

Nota. Generalizzazione dell'abbinamento bipartito.

Abbinamento tridimensionale

3D-MATCHING. Dati 3 insiemi disgiunti X, Y, e Z, ognuno di taglia n ed un insieme $T \subseteq X \times Y \times Z$ di triple, esiste un insieme di n triple di T tale che ogni elemento di $X \cup Y \cup Z$ appartiene ad esattamente una di quelle triple?

Teorema. 3-SAT $\leq_P$ 3D-MATCHING.

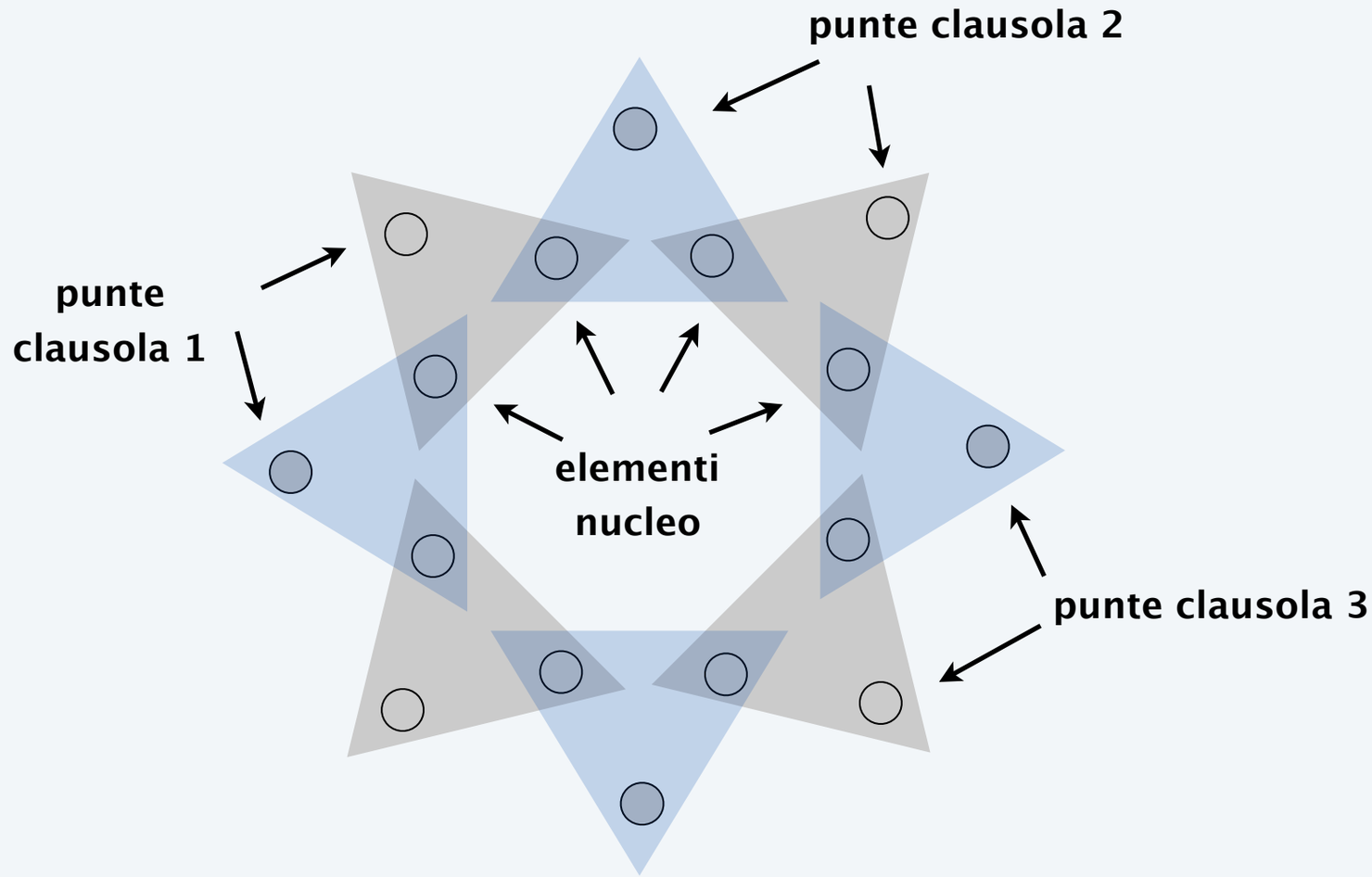
Dim. Data un'istanza Φ di 3-SAT, costruiamo un'istanza di 3D-MATCHING che ha un abbinamento perfetto sse Φ è soddisfacibile.

3-Satisfiability è riducibile a 3D-Matching

Costruzione. (parte 1)

k = numero di clausole

- Crea un gadget per ogni variabile x_i con $2k$ elementi "nucleo" e $2k$ elementi "punta".



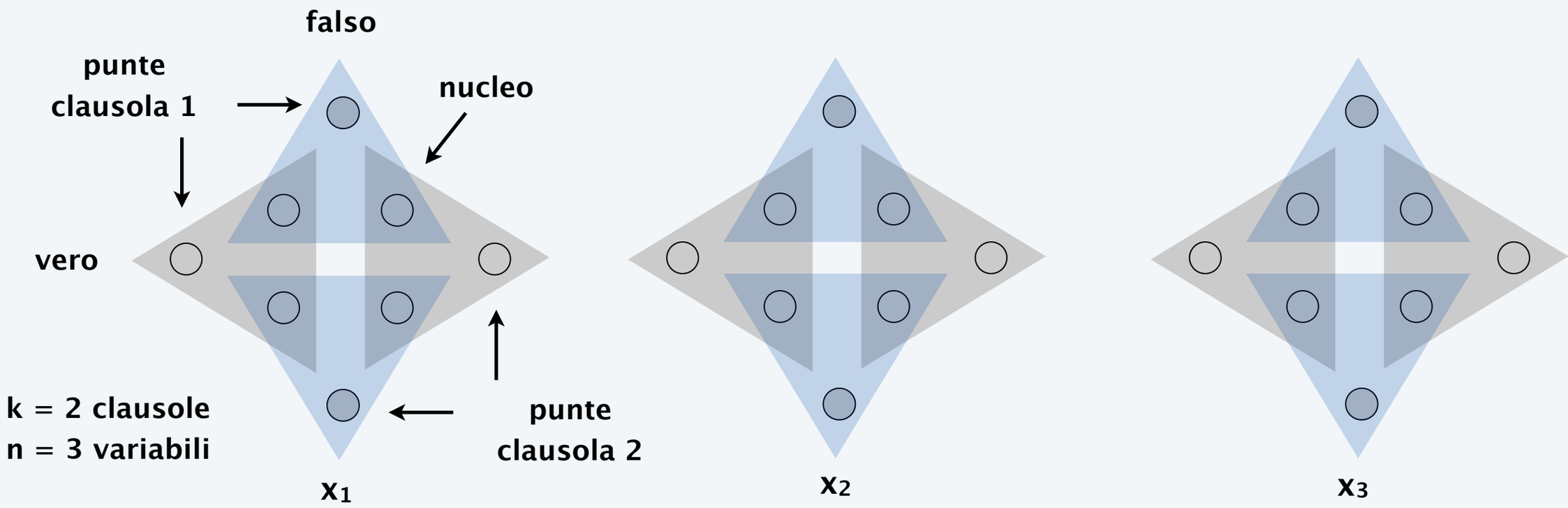
un gadget per la variabile x_i (k = 4)

3-Satisfiability è riducibile a 3D-Matching

Costruzione. (parte 1)

k = numero di clausole

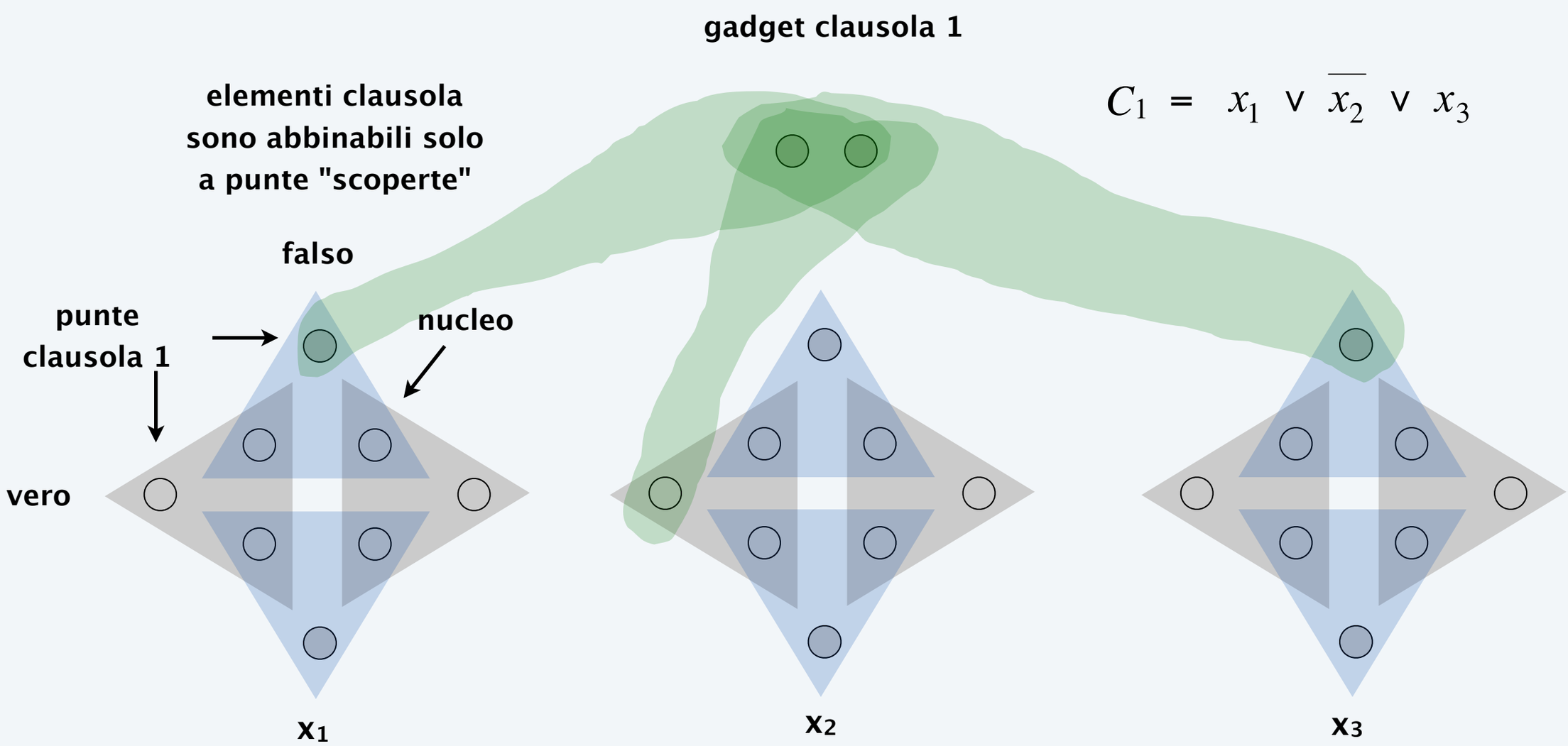
- Crea un gadget per ogni variabile x_i con $2k$ elementi "nucleo" e $2k$ elementi "punta".
- Nessun'altra tripla utilizza gli elementi "nucleo".
- Nel gadget per x_i, ogni abbinamento perfetto deve utilizzare o tutte le triple grigie (corrispondenti a $x_i = true$) o tutte quelle blu (corrispondenti a $x_i = false$).



3-Satisfiability è riducibile a 3D-Matching

Costruzione. (parte 2)

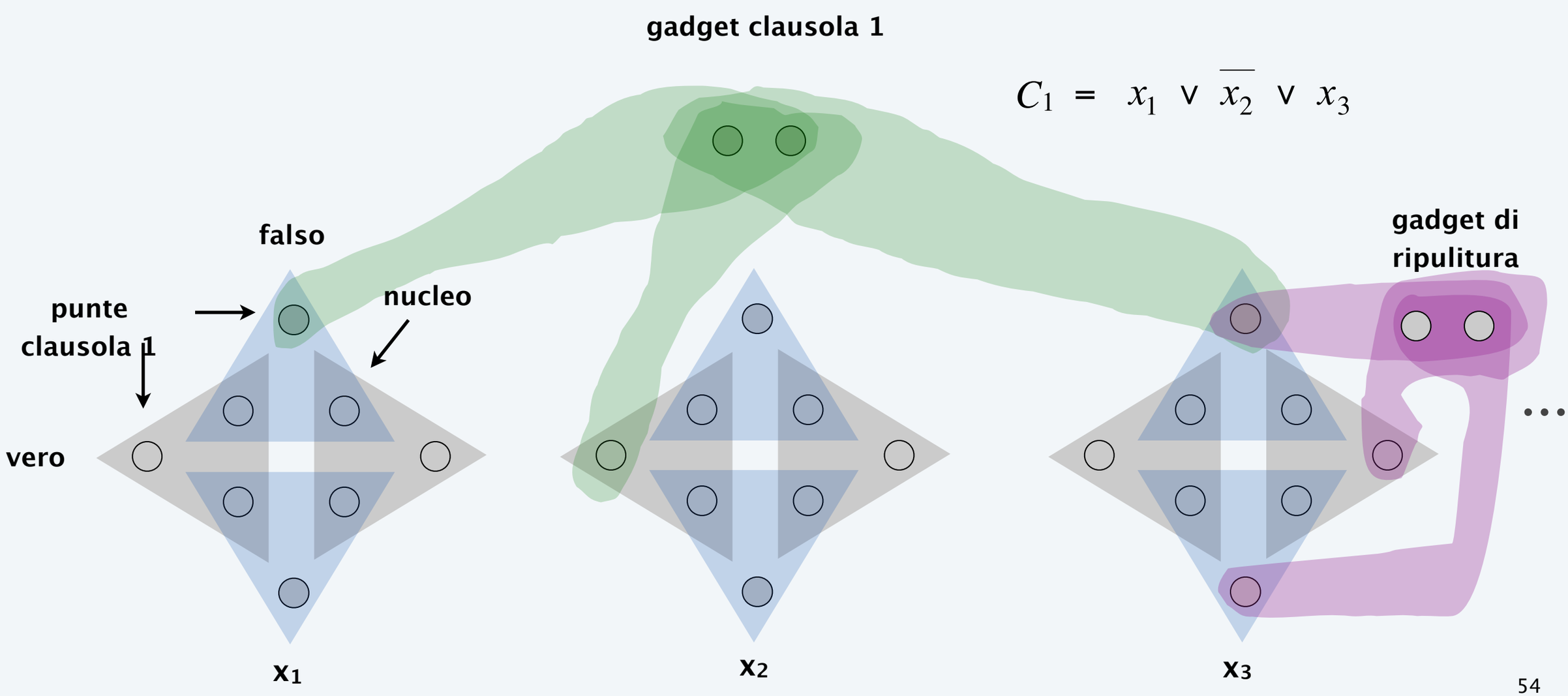
- Crea un gadget per ogni clausola C_j con due elementi e tre triple.
- Esattamente una di queste triple sarà usata in un abbinamento perfetto.
- Nell'esempio, ciò assicura che ogni abbinamento perfetto usa o (i) nucleo grigio di x_1 o (ii) nucleo blu di x_2 o (iii) nucleo grigio di x_3.



3-Satisfiability è riducibile a 3D-Matching

Costruzione. (parte 3)

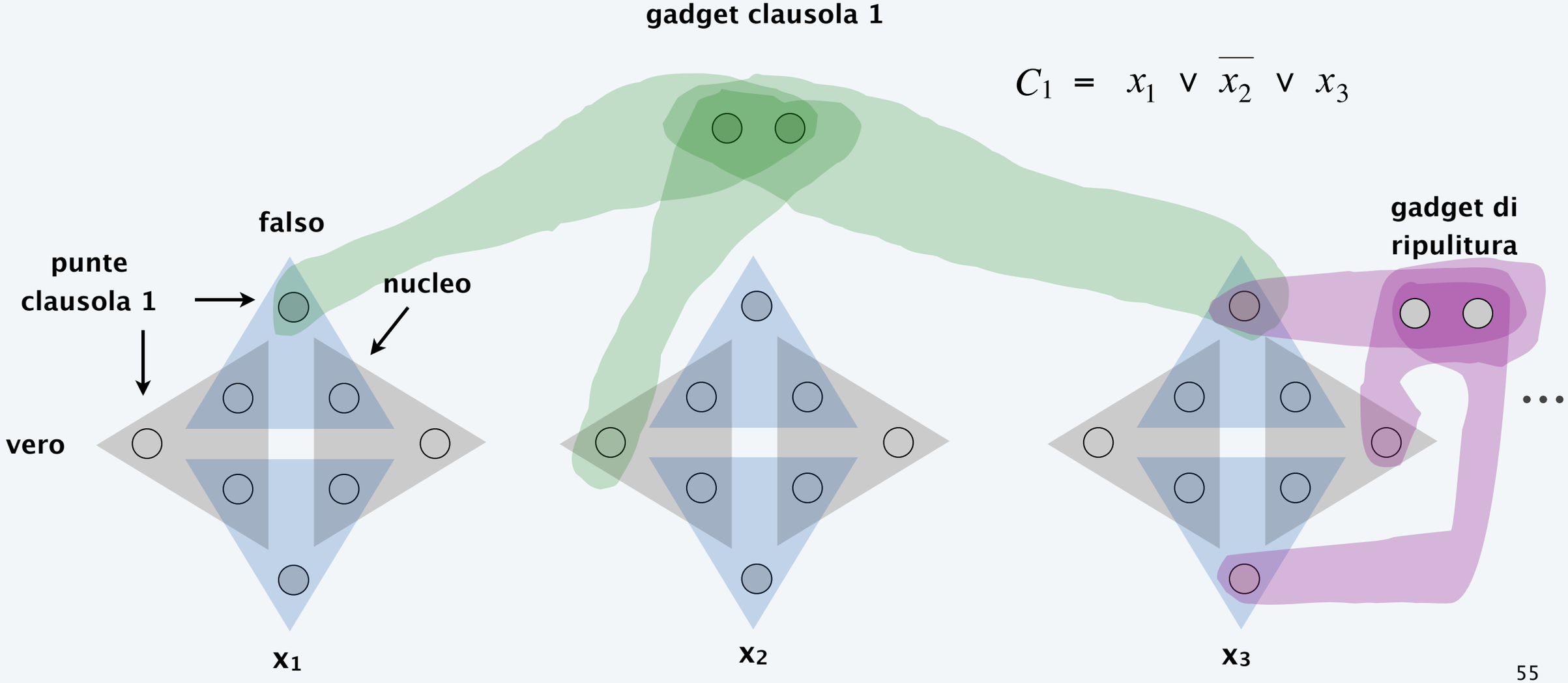
- Ci sono $2nk$ punte: nk coperte da triple blu/grigie; k da triple clausola.
- Per coprire le restanti $(n-1)k$ punte, crea $(n-1)k$ gadget di "ripulitura": simili ai gadget clausola, ma con $2nk$ triple, connesse ad ogni punta.



3-Satisfiability è riducibile a 3D-Matching

Lemma. L'istanza (X, Y, Z) ha un abbinamento perfetto sse Φ è soddisfacibile.

D. Chi sono X, Y, e Z?

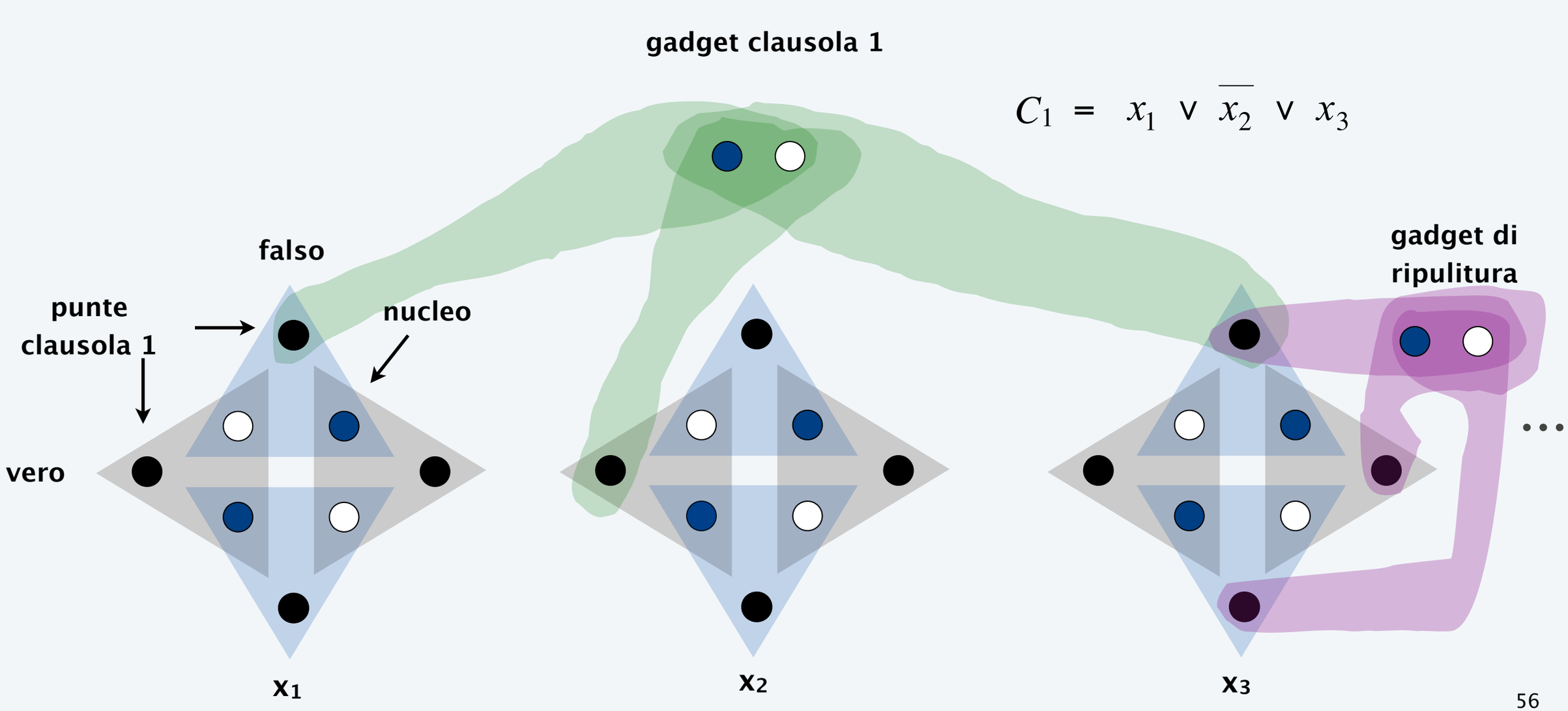


3-Satisfiability è riducibile a 3D-Matching

Lemma. L'istanza (X, Y, Z) ha un abbinamento perfetto sse Φ è soddisfacibile.

D. Chi sono X, Y, e Z?

R. $X = nero$, $Y = bianco$, e $Z = blu$.

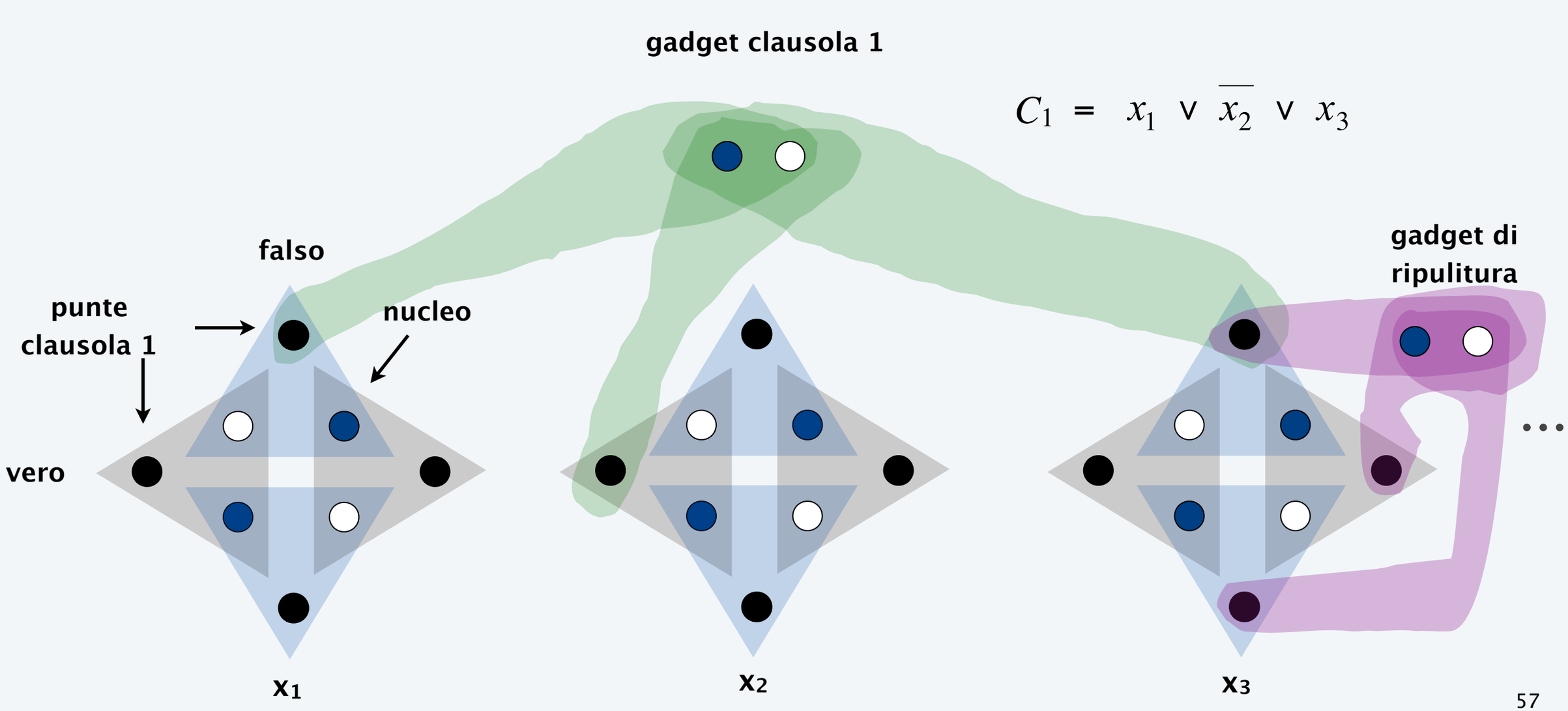


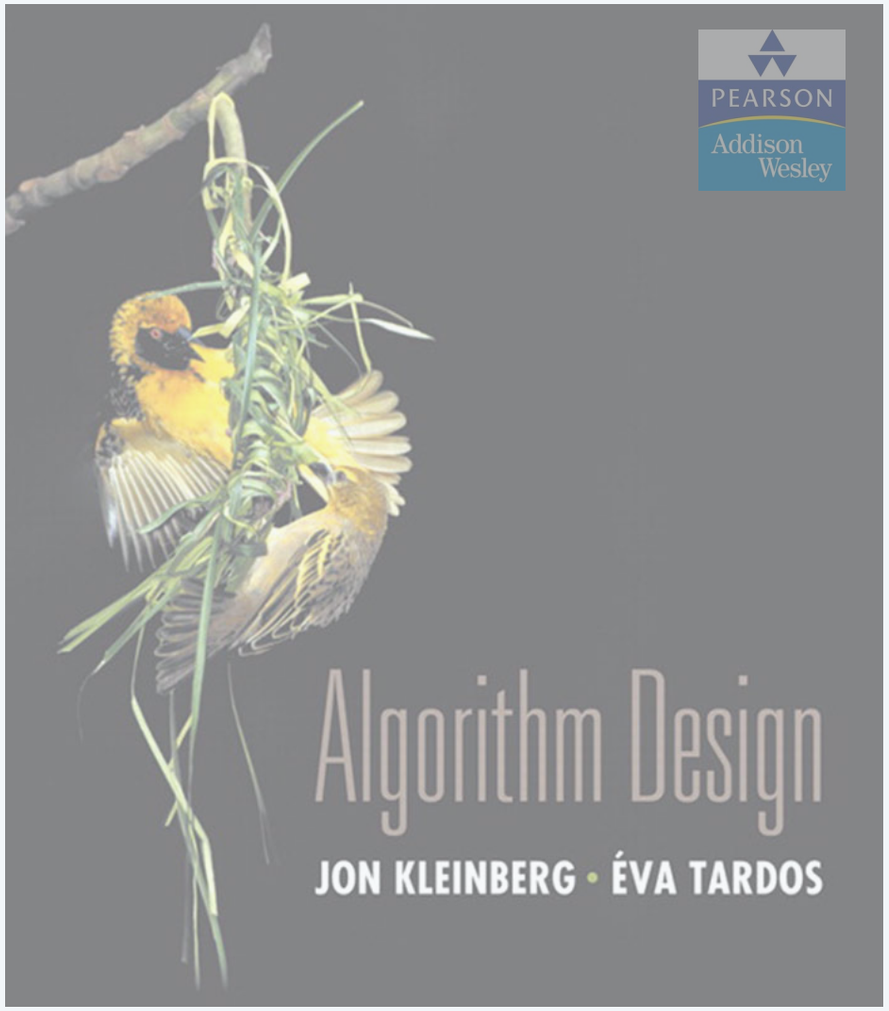
3-Satisfiability è riducibile a 3D-Matching

Lemma. L'istanza (X, Y, Z) ha un abbinamento perfetto sse Φ è soddisfacibile.

Dim. $\Rightarrow$ Se un 3d-matching esiste, imposta x_i in base al gadget di x_i.

Dim. $\Leftarrow$ Se Φ è soddisfacibile, usa un qualunque letterale vero di C_j per scegliere una tripla del gadget C_j. ▪





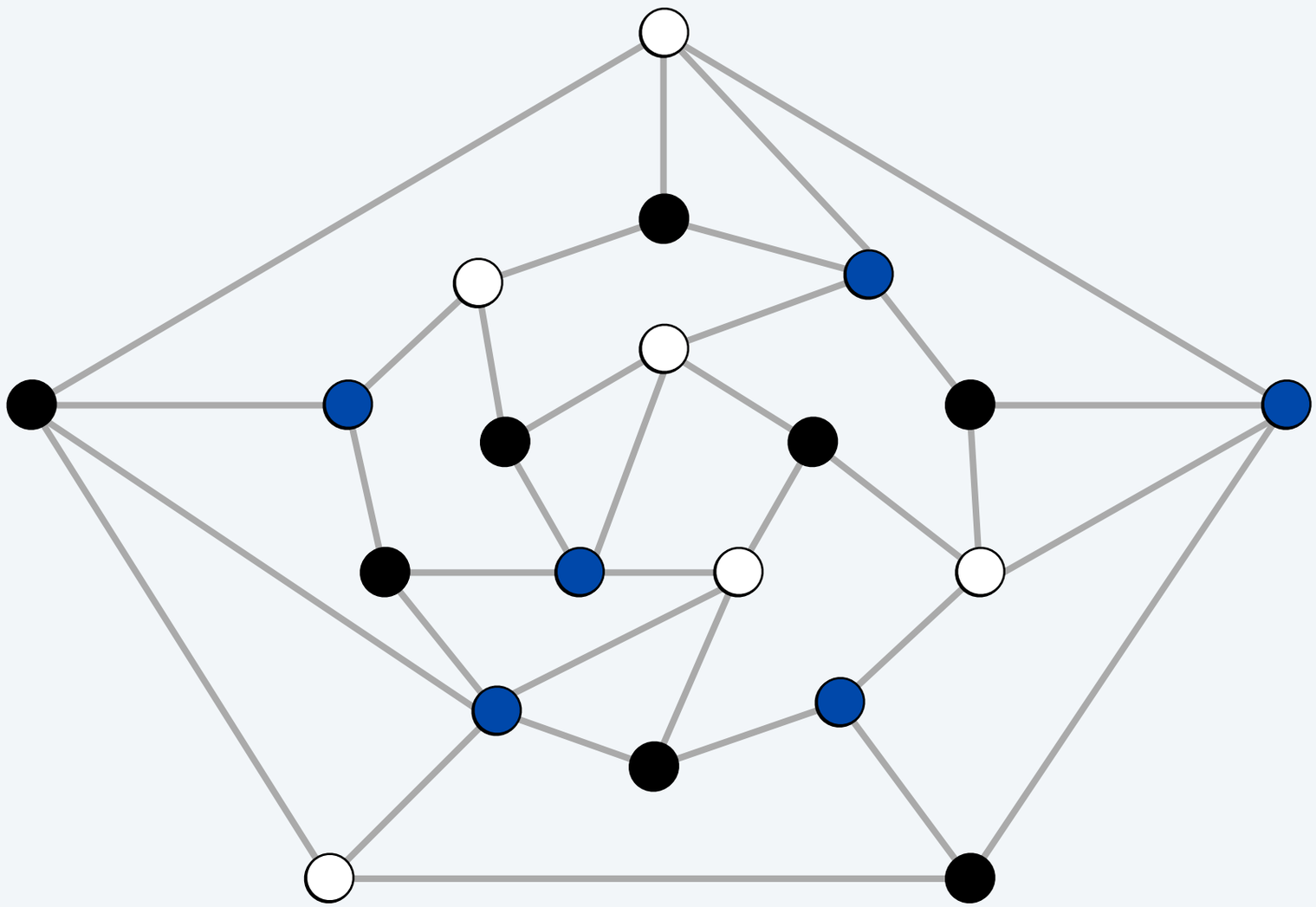
SECTION 8.7

8. INTRATTABILITÀ I

- *poly-time reductions*
- *packing and covering problems*
- *constraint satisfaction problems*
- *sequencing problems*
- *partitioning problems*
- *colorazione di grafi*
- *numerical problems*

3-Colorabilità [3-Coloring]

3-COLOR. Dato un grafo non orientato G, possiamo colorarne i nodi di nero, bianco e blu in modo tale che nessuna coppia di nodi adiacenti abbia lo stesso colore?



istanza yes

Intrattabilità: quiz 6

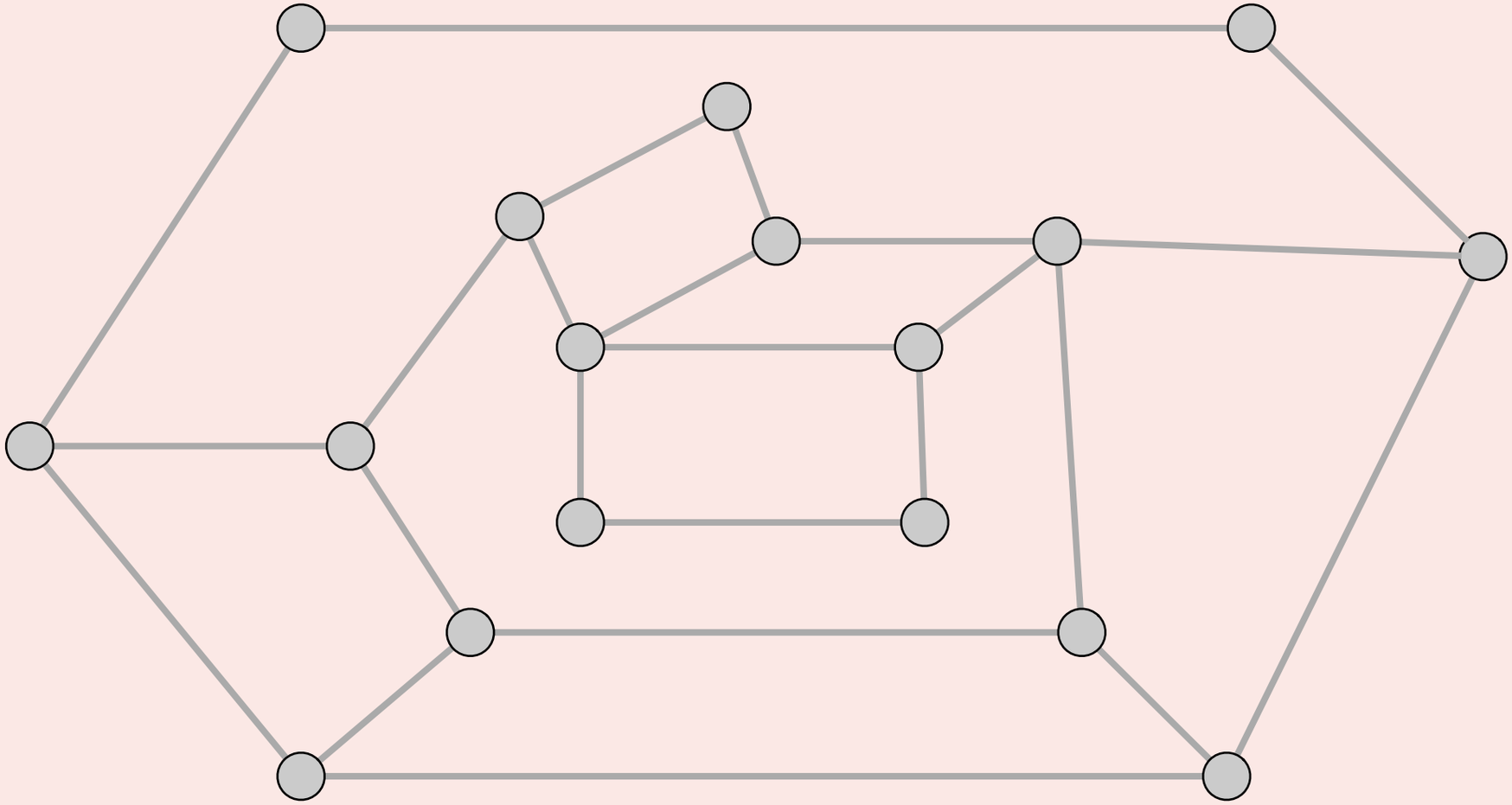
Quanto è difficile risolvere 2-COLOR?

A. $O(m + n)$ tramite BFS o DFS.

B. $O(mn)$ tramite massimo flusso.

C. $\Omega(2^n)$ tramite forza bruta.

D. Neanche Tarjan lo sa.



Applicazione: allocazione di registri

Allocazione di registri. Assegnare variabili di un programma ai registri del calcolatore in modo da usare al più k registri e in modo che non ci siano coppie di variabili richieste allo stesso momento assegnate allo stesso registro.

Grafo di interferenza. I nodi sono le variabili del programma; c'è un arco tra u e v se esiste un'operazione in cui sia u che v sono "accessibili" nello stesso momento.

Osservazione. [Chaitin 1982] L'allocazione dei registri è possibile sse il grafo di interferenza è k-colorabile.

Fatto. 3-COLOR $\leq_P$ K-REGISTER-ALLOCATION per ogni costante $k \geq 3$.

REGISTER ALLOCATION & SPILLING VIA GRAPH COLORING

G. J. Chaitin
IBM Research
P.O.Box 218, Yorktown Heights, NY 10598

3-Satisfiability è riducibile a 3-Coloring

Teorema. 3-SAT $\leq_P$ 3-COLOR.

Dim. Data un'istanza di 3-SAT Φ, costruiamo un'istanza di 3-COLOR che è 3-colorabile sse Φ è soddisfacibile.

3-Satisfiability è riducibile a 3-Coloring

Costruzione.

(i) Crea un grafo G con un nodo per ogni letterale.

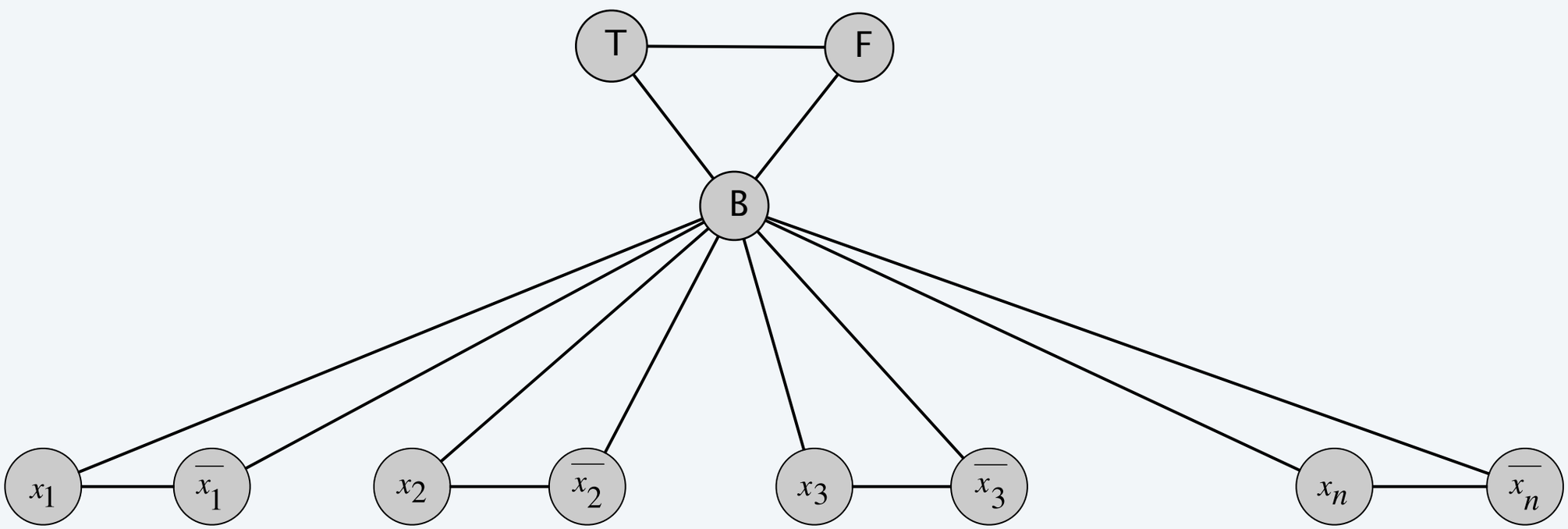
(ii) Collega ogni letterale al suo negato.

(iii) Crea 3 nuovi nodi T, F, e B; collegali formando un triangolo.

(iv) Collega ciascun letterale a B.

(v) Per ogni clausola C_j, aggiungi un gadget di 6 nodi e 13 archi.

descritto oltre

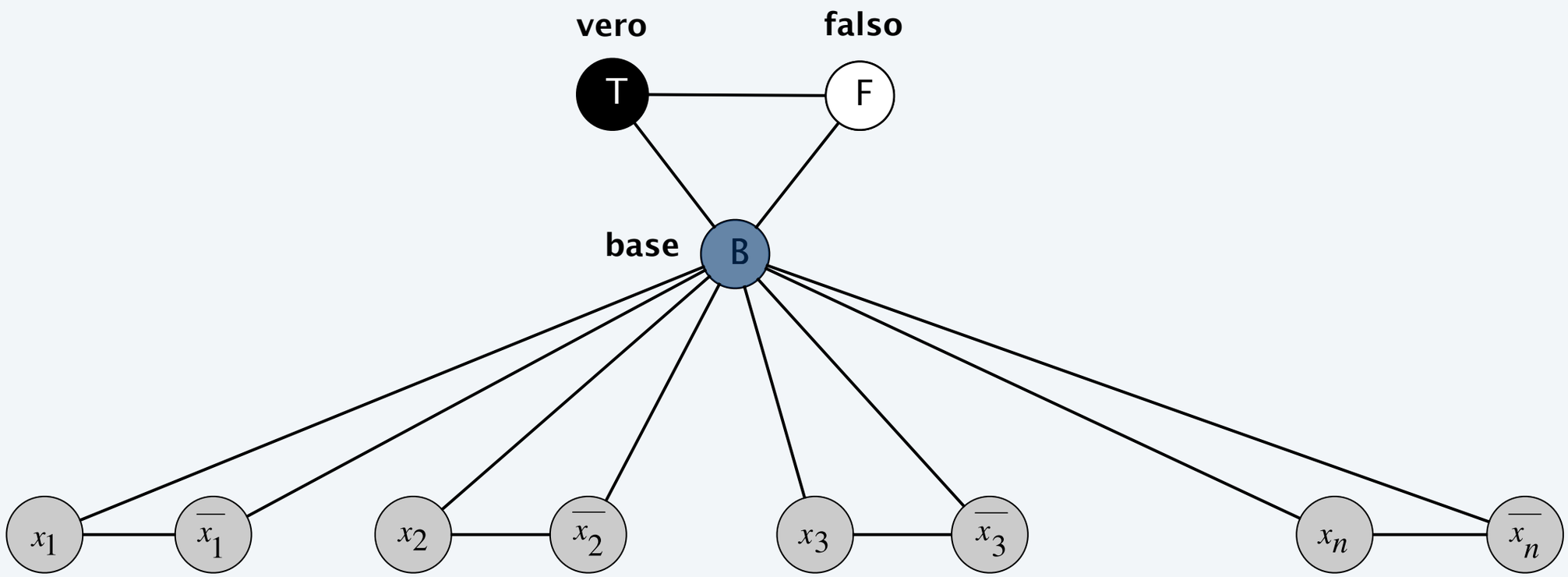


3-Satisfiability è riducibile a 3-Coloring

Lemma. Il grafo G è 3-colorabile sse Φ è soddisfacibile.

Dim. $\Rightarrow$ Supponi che il grafo G sia 3-colorabile.

- WLOG, sia il nodo T colorato di *nero*, F di *bianco*, e B di *blue*.
- Considera assegnazione che mette i letterali *neri* a *true* (e *bianchi* a *false*).
- (iv) fa sì che ogni letterale sia colorato *nero* o *bianco*.
- (ii) fa sì che ogni letterale sia *bianco* se il suo negato è *nero* (e viceversa).

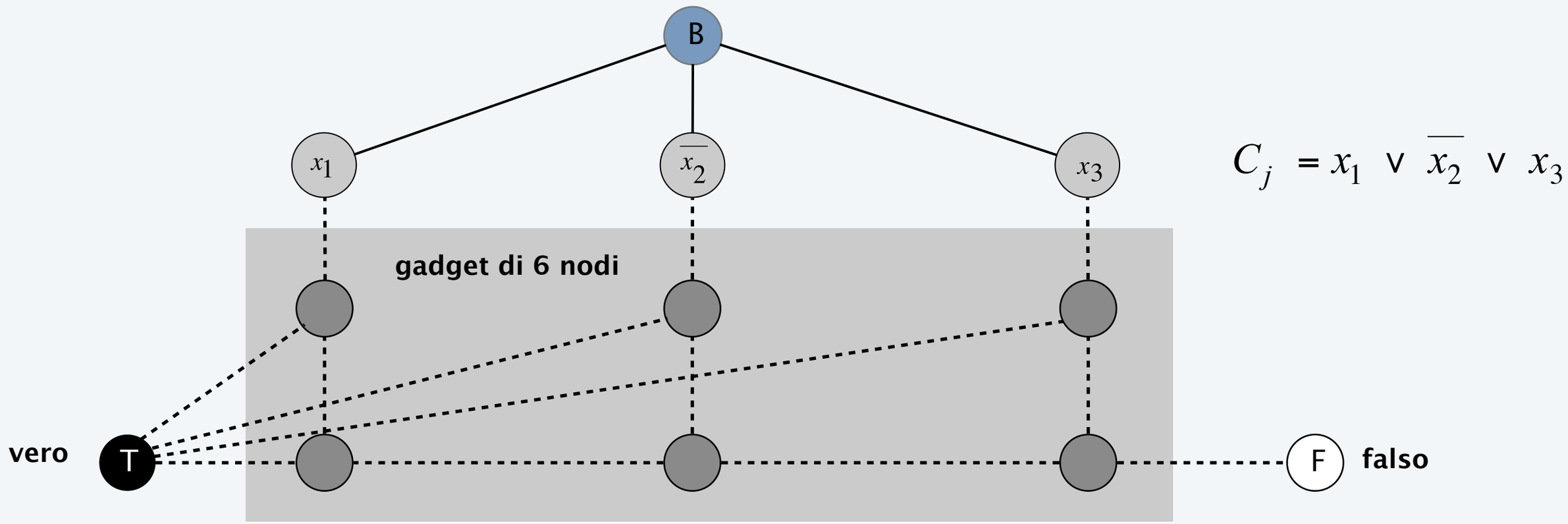


3-Satisfiability è riducibile a 3-Coloring

Lemma. Il grafo G è 3-colorabile sse Φ è soddisfacibile.

Dim. $\Rightarrow$ Supponi che il grafo G sia 3-colorabile.

- WLOG, sia il nodo T colorato di *nero*, F di *bianco*, e B di *blue*.
- Considera assegnazione che mette i letterali *neri* a *true* (e *bianchi* a *false*).
- (iv) fa sì che ogni letterale sia colorato *nero* o *bianco*.
- (ii) fa sì che ogni letterale sia *bianco* se il suo negato è *nero* (e viceversa)
- (v) fa sì che almeno un letterale di ogni clausola sia *nero*.

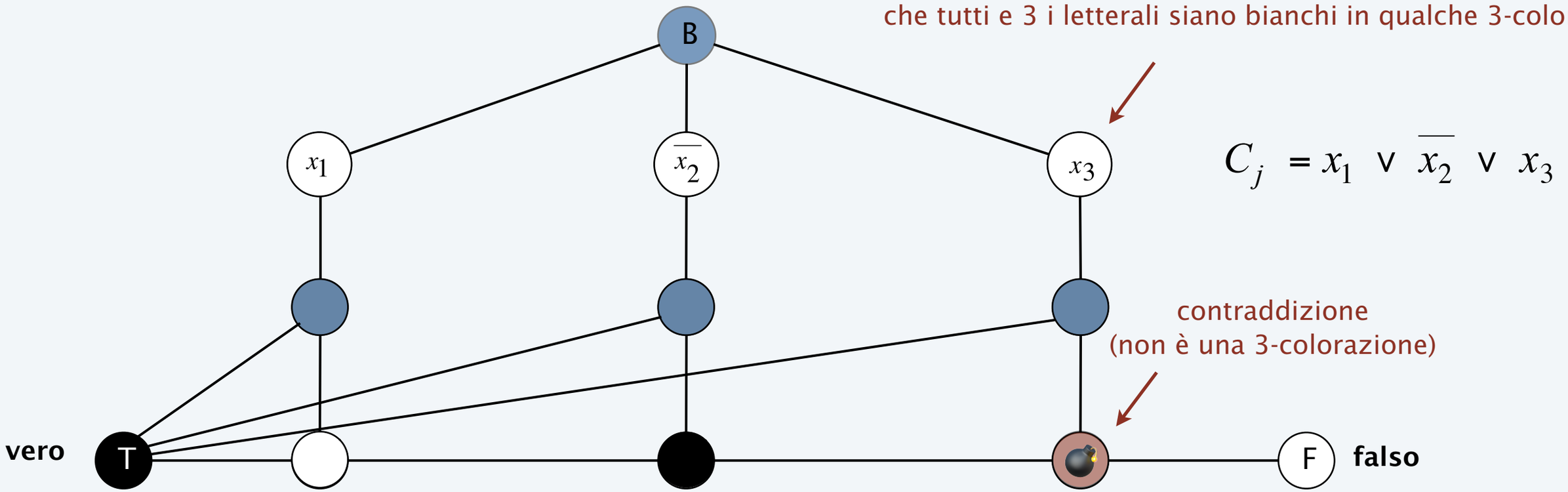


3-Satisfiability è riducibile a 3-Coloring

Lemma. Il grafo G è 3-colorabile sse Φ è soddisfacibile.

Dim. $\Rightarrow$ Supponi che il grafo G sia 3-colorabile.

- WLOG, sia il nodo T colorato di *nero*, F di *bianco*, e B di *blue*.
- Considera assegnazione che mette i letterali *neri* a *true* (e *bianchi* a *false*).
- (iv) fa sì che ogni letterale sia colorato *nero* o *bianco*.
- (ii) fa sì che ogni letterale sia *bianco* se il suo negato è *nero* (e viceversa)
- (v) fa sì che almeno un letterale di ogni clausola sia *nero*.

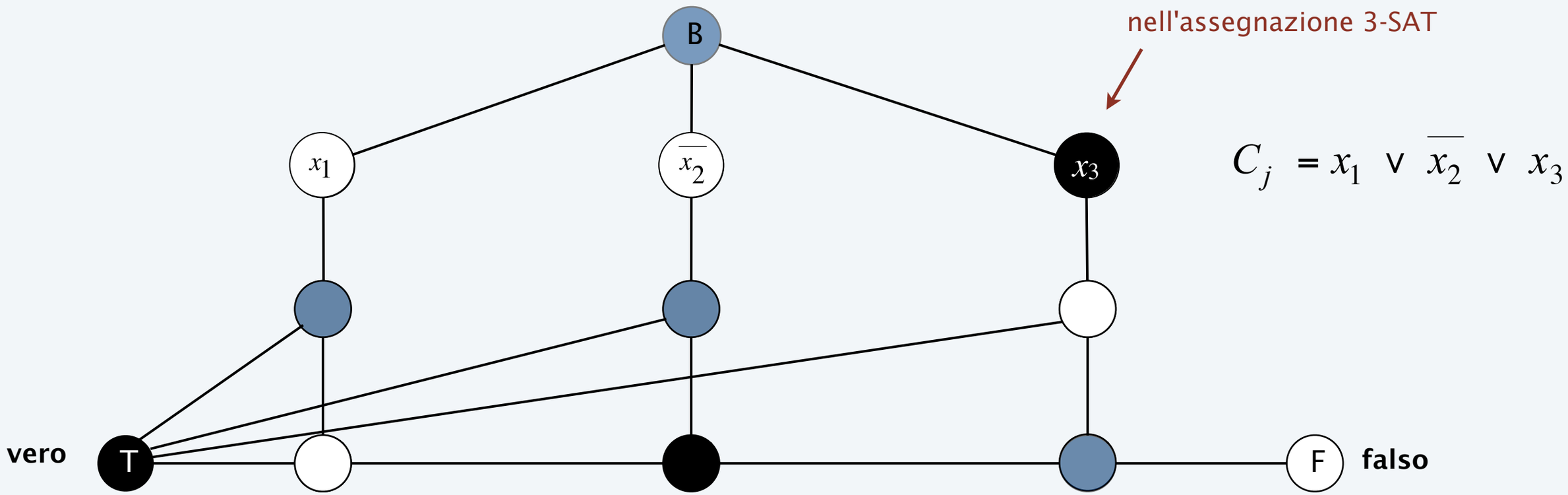


3-Satisfiability è riducibile a 3-Coloring

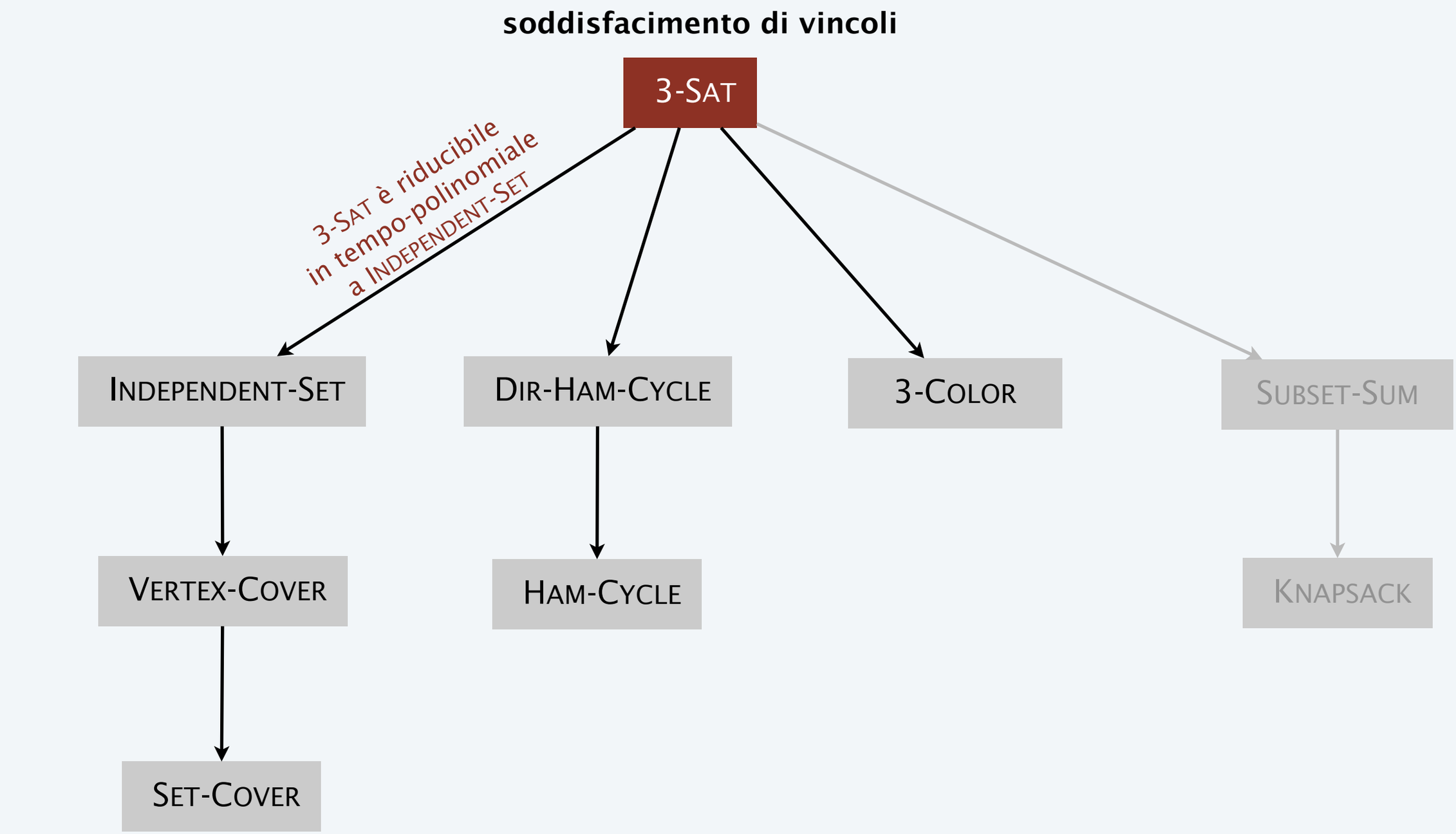
Lemma. Il grafo G è 3-colorabile sse Φ è soddisfacibile.

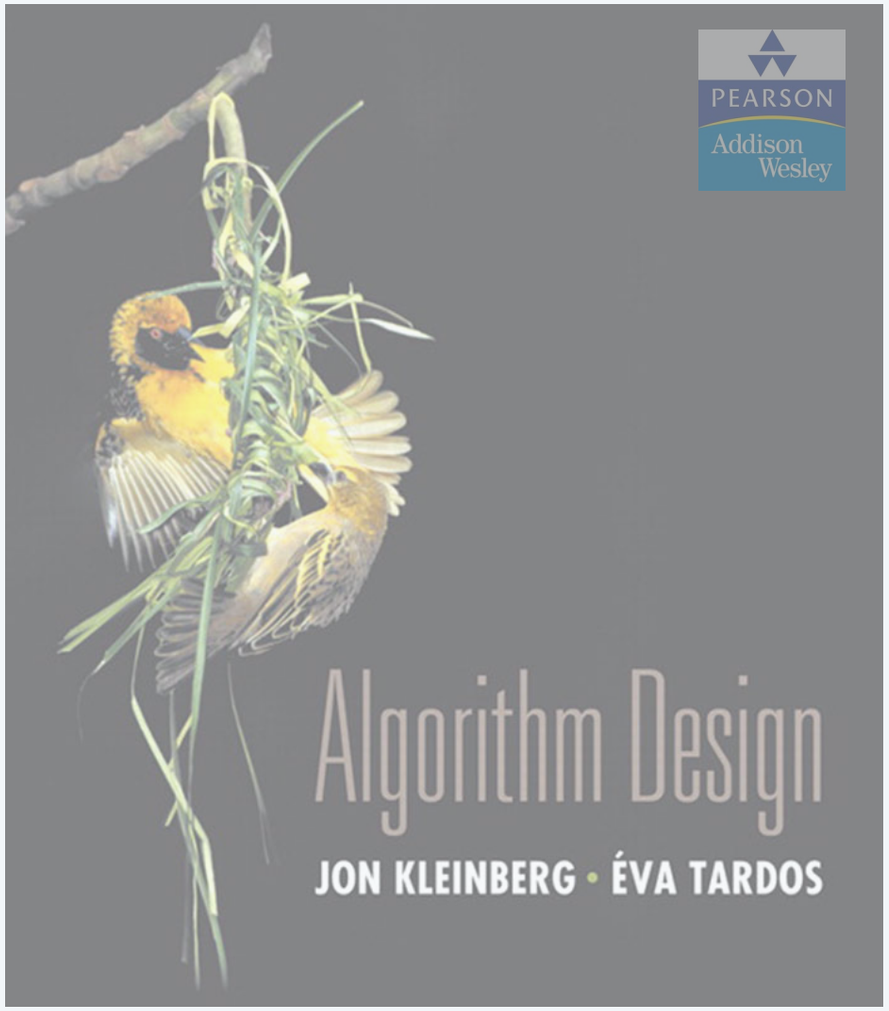
Dim. $\Leftarrow$ Supponi che l'istanza 3-SAT Φ sia soddisfacibile.

- Colora tutti i letterali *true* di *nero* e quelli *false* di *bianco*.
- Scegli un letterale *true*; colora il nodo sottostante di *bianco*, e quello sottostante al sottostante di *blu*.
- Colora gli altri nodi della riga di mezzo di *blu*.
- Colora gli altri nodi dell'ultima riga di *nero* o *bianco*, come costretto. ▪



Riduzioni tempo-polinomiali



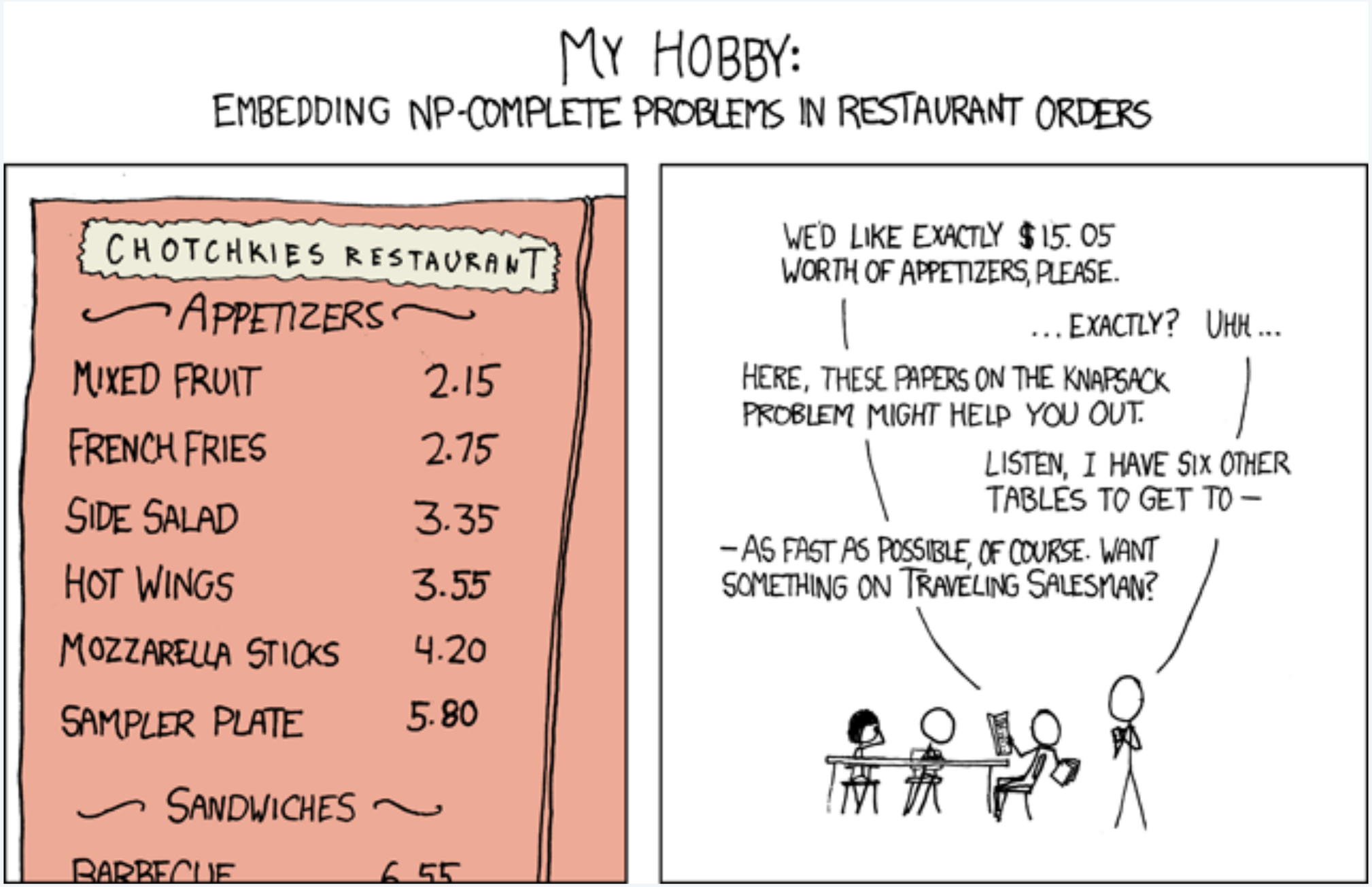


SECTION 8.8

8. INTRATTABILITÀ I

- *poly-time reductions*
- *packing and covering problems*
- *constraint satisfaction problems*
- *sequencing problems*
- *partitioning problems*
- *graph coloring*
- *problemi numerici*

My hobby



Somma di sottoinsieme [Subset sum]

SUBSET-SUM. Dati n numeri naturali $w_1, \ldots, w_n$ ed un intero W, esiste un sottoinsieme dei numeri dati la cui somma è esattamente W?

Es. $\{ 215, 215, 275, 275, 355, 355, 420, 420, 580, 580, 655, 655 \}$, $W = 1505$.

Sì. $215 + 355 + 355 + 580 = 1505$.

Nota. Nei problemi aritmetici, gli interi in input sono codificati in binario. Le riduzioni tempo-polinomiali devono essere polinomiali rispetto alla codifica binaria.

Somma di sottoinsieme

Teorema. 3-SAT $\leq_P$ SUBSET-SUM.

Dim. Data un'istanza Φ di 3-SAT, costruiamo un'istanza di SUBSET-SUM che ha soluzione sse Φ è soddisfacibile.

3-Satisfiability è riducibile a Subset Sum

Costruzione. Data un'istanza 3-SAT Φ con n variabili e k clausole, formiamo $2n + 2k$ interi decimali, ognuno con $n + k$ cifre:

- Includi una cifra per ogni variabile x_i ed una per ogni clausola C_j.
- Includi due numeri per ogni variabile x_i.
- Includi due numeri per ogni clausola C_j.
- La somma di ogni cifra x_i deve fare 1; la somma di ogni cifra C_j deve fare 4.

Proprietà chiave. Non sono possibili riporti

$\Rightarrow$ ogni cifra fornisce una equazione.

$$
\begin{aligned}
C_1 &= \neg x_1 \vee x_2 \vee x_3 \\
C_2 &= x_1 \vee \neg x_2 \vee x_3 \\
C_3 &= \neg x_1 \vee \neg x_2 \vee \neg x_3
\end{aligned}
$$

istanza 3-SAT

	x_1	x_2	x_3	C_1	C_2	C_3	
x_1	1	0	0	0	1	0	100.010
$\neg\, x_1$	1	0	0	1	0	1	100.101
x_2	0	1	0	1	0	0	10.100
$\neg\, x_2$	0	1	0	0	1	1	10.011
x_3	0	0	1	1	1	0	1.110
$\neg\, x_3$	0	0	1	0	0	1	1.001
	0	0	0	1	0	0	100
	0	0	0	2	0	0	200
	0	0	0	0	1	0	10
	0	0	0	0	2	0	20
	0	0	0	0	0	1	1
	0	0	0	0	0	2	2
W	1	1	1	4	4	4	111.444

elementi fantoccio per far sommare le clausole colonna a 4

istanza SUBSET-SUM

3-Satisfiability è riducibile a Subset Sum

Lemma. Φ è soddisfacibile sse esiste un sottoinsieme con somma W.

Dim. $\Rightarrow$ Supponi che l'istanza 3-SAT Φ abbia un'assegnazione valida x^*.

- Se $x_i^* = true$, scegli l'intero della riga x_i ;
  altrimenti, scegli l'intero della riga $\neg\, x_i$.
- Ogni cifra x_i somma ad 1.
- Poiché Φ è soddisfacibile, ogni cifra C_j ha somma ≥ 1 dalle righe x_i e $\neg\, x_i$.
- Scegli interi fantoccio per portare la somma di ogni cifra C_j a 4. ▪

$$C_1 = \neg\, x_1 \vee x_2 \vee x_3$$
$$C_2 = x_1 \vee \neg\, x_2 \vee x_3$$
$$C_3 = \neg\, x_1 \vee \neg\, x_2 \vee \neg\, x_3$$

istanza 3-SAT

elementi fantoccio per far sommare le clausole colonna a 4

	x_1	x_2	x_3	C_1	C_2	C_3	
x_1	1	0	0	0	1	0	100.010
$\neg\, x_1$	1	0	0	1	0	1	100.101
x_2	0	1	0	1	0	0	10.100
$\neg\, x_2$	0	1	0	0	1	1	10.011
x_3	0	0	1	1	1	0	1.110
$\neg\, x_3$	0	0	1	0	0	1	1.001
	0	0	0	1	0	0	100
	0	0	0	2	0	0	200
	0	0	0	0	1	0	10
	0	0	0	0	2	0	20
	0	0	0	0	0	1	1
	0	0	0	0	0	2	2
W	1	1	1	4	4	4	111.444

istanza SUBSET-SUM

3-Satisfiability è riducibile a Subset Sum

Lemma. Φ è soddisfacibile sse esiste un sottoinsieme con somma W.

Dim. $\Leftarrow$ Supponi che esista un sottoinsieme S^* che ha somma W.

- La cifra x_i costringe a scegliere o la riga x_i o la riga $\neg\, x_i$ (e non entrambe).
- Se è scelta la riga x_i, poni $x_i^* = true$; altrimenti, poni $x_i^* = false$.

  La cifra C_j costringe a scegliere almeno un letterale vero per ogni clausola. ▪

$$C_1 = \neg\, x_1 \vee x_2 \vee x_3$$
$$C_2 = x_1 \vee \neg\, x_2 \vee x_3$$
$$C_3 = \neg\, x_1 \vee \neg\, x_2 \vee \neg\, x_3$$

istanza 3-SAT

elementi fantoccio per far sommare le clausole colonna a 4

	x_1	x_2	x_3	C_1	C_2	C_3	
x_1	1	0	0	0	1	0	100.010
$\neg\, x_1$	1	0	0	1	0	1	100.101
x_2	0	1	0	1	0	0	10.100
$\neg\, x_2$	0	1	0	0	1	1	10.011
x_3	0	0	1	1	1	0	1.110
$\neg\, x_3$	0	0	1	0	0	1	1.001
	0	0	0	1	0	0	100
	0	0	0	2	0	0	200
	0	0	0	0	1	0	10
	0	0	0	0	2	0	20
	0	0	0	0	0	1	1
	0	0	0	0	0	2	2
W	1	1	1	4	4	4	111.444

istanza SUBSET-SUM

SUBSET SUM È RIDUCIBILE A KNAPSACK

SUBSET-SUM. Dati n numeri naturali $w_1, \ldots, w_n$ ed un intero W, esiste un sottoinsieme che ha somma esattamente W?

KNAPSACK. Dato un insieme di elementi X, pesi $u_i \geq 0$, valori $v_i \geq 0$, un limite di peso U, e un valore bersaglio V, esiste un sottoinsieme $S \subseteq X$ tale che:

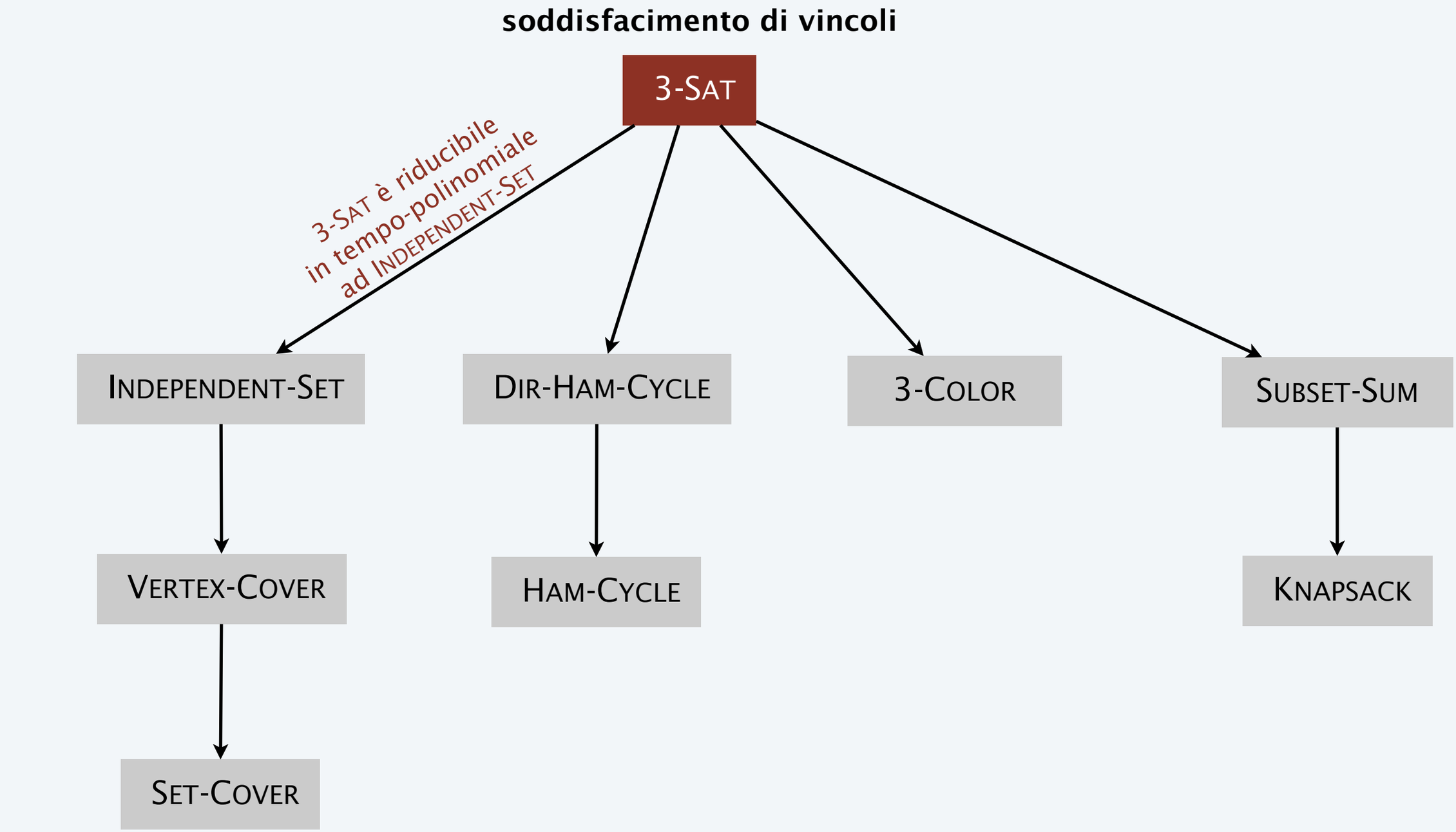
$$\sum_{i \in S} u_i \leq U, \quad \sum_{i \in S} v_i \geq V$$

Ricordiamo. Algoritmo di programmazione dinamica $O(n\,U)$ per KNAPSACK.

Esercizio. Dimostrare che SUBSET-SUM $\leq_{\mathrm{P}}$ KNAPSACK.

Dim. Data l'istanza $(w_1, \ldots, w_n, W)$ di SUBSET-SUM, crea l'istanza KNAPSACK:

Riduzioni tempo-polinomiali



Le 20 riduzioni tempo-polinomiali di Karp da Satisfiability

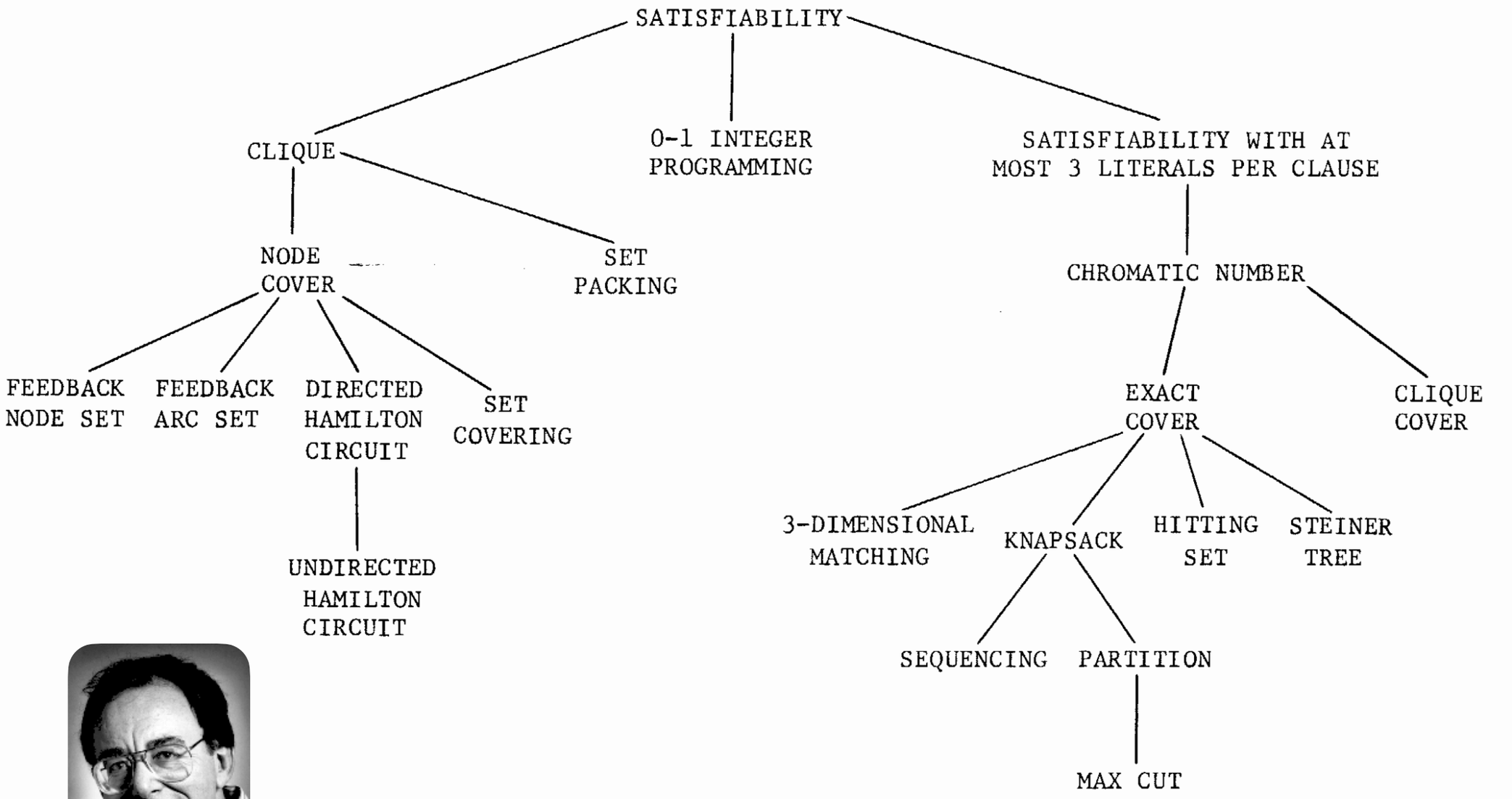


FIGURE 1 - Complete Problems

RICHARD M. KARP

Dick Karp (1972)
Premio Turing 1985