Remarks

- We had proved that MAX CUT is NP-complete for multigraphs.
- How about proving the same thing for simple graphs?[a]
- For 4SAT, how do you modify the proof?[b]
- All NP-complete problems are mutually reducible by definition as an NP-complete problem is in NP.[c]
  - So they are equally hard in this sense.[d]

[a]Contributed by Mr. Tai-Dai Chou (J93922005) on June 2, 2005.
[b]Contributed by Mr. Chien-Lin Chen (J94922015) on June 8, 2006.
[c]Contributed by Mr. Ren-Shuo Liu (D98922016) on October 27, 2009.
[d]Contributed by Mr. Ren-Shuo Liu (D98922016) on October 27, 2009.

MAX BISECTION

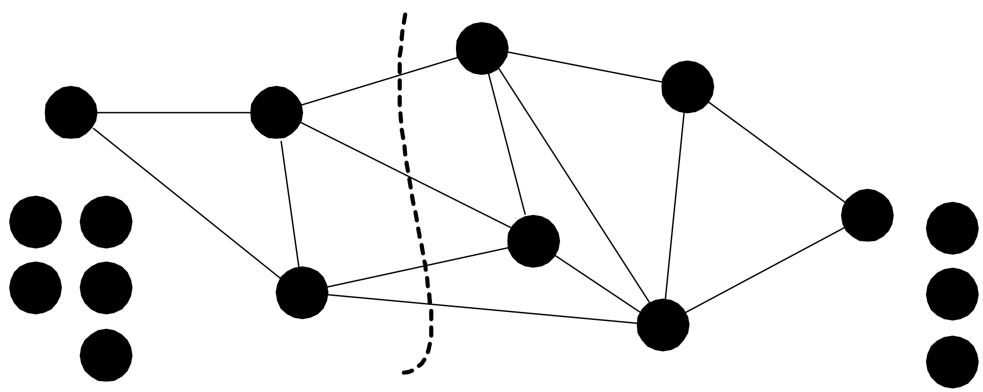
- MAX CUT becomes MAX BISECTION if we require that $|S| = |V - S|$.
- It has many applications, especially in VLSI layout.

MAX BISECTION Is NP-Complete

- We shall reduce the more general MAX CUT to MAX BISECTION.
- Add $|V| = n$ **isolated nodes** to G to yield G'.
- G' has $2n$ nodes.
- As the new nodes have no edges, moving them around contributes nothing to the cut.

The Proof (concluded)

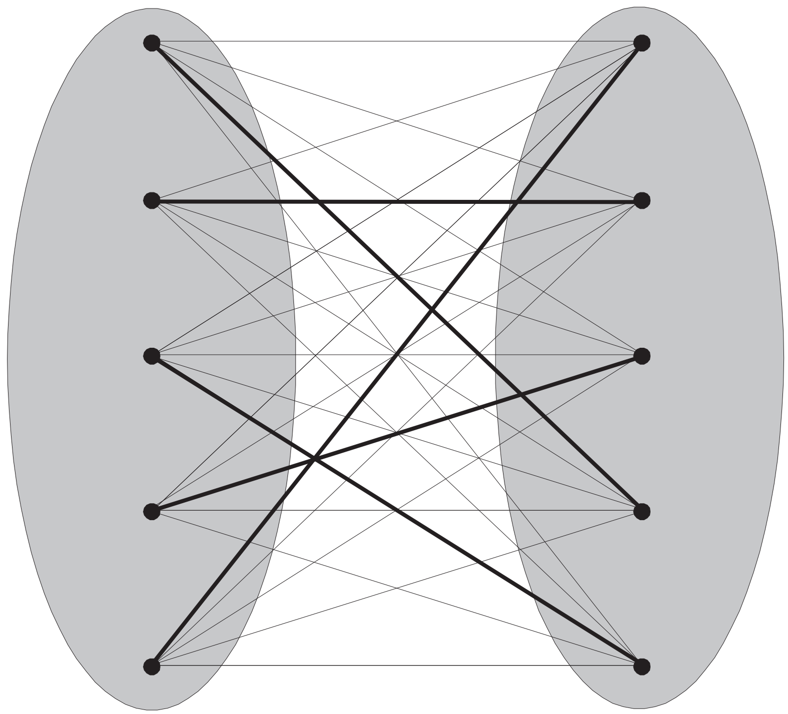
- Every cut $(S, V - S)$ of $G = (V, E)$ can be made into a bisection by appropriately allocating the new nodes between S and $V - S$.
- Hence each cut of G can be made a cut of G' of the same size, and vice versa.



BISECTION WIDTH

- BISECTION WIDTH is like MAX BISECTION except that it asks if there is a bisection of size *at most* K (sort of MIN BISECTION).
- Unlike MIN CUT, BISECTION WIDTH remains NP-complete.
  - A graph $G = (V, E)$, where $|V| = 2n$, has a bisection of size K if and only if the complement of G has a bisection of size $n^2 - K$.
  - So G has a bisection of size $\geq K$ if and only if its complement has a bisection of size $\leq n^2 - K$.

Illustration



HAMILTONIAN PATH Is NP-Complete[a]

Theorem 42 *Given an* undirected *graph, the question whether it has a Hamiltonian path is NP-complete.*

- We will reduce 3SAT to HAMILTONIAN PATH.
- We are given a boolean expression $\phi(x_1, x_2, \ldots, x_n)$ in CNF.
- The clauses are $C_1, C_2, \ldots, C_m$, each containing 3 literals.
- Need to construct a graph $R(\phi)$ that has a Hamiltonian path if and only if $\phi \in$ 3SAT.

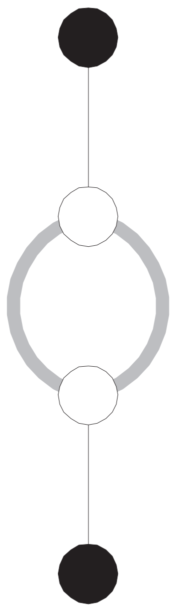
[a]Karp (1972).

The Proof (continued)

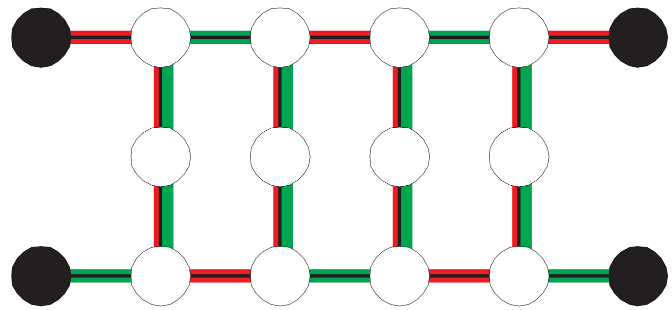
- Each boolean variable must be either true or false ("choice").
- We need to impose that all occurrences of x be assigned the same truth value ("consistency").
- We must also make sure that all occurrences of $\neg x$ be assigned the opposite truth value ("consistency").
- Finally, the clauses provide the constraints that must be satisfied in 3SAT ("constraint").

The Proof (continued)

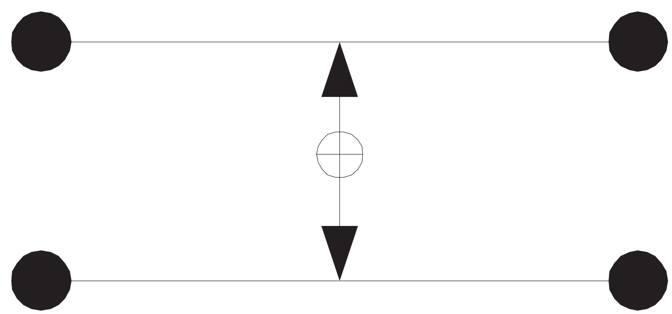
- Both the choice gadgets and the consistency gadgets will be used to build $R(\phi)$.
- The choice gadget makes sure that a Hamiltonian path must take either the left parallel edge (true) or the right parallel edge (false).



Choice gadget



Consistency gadget

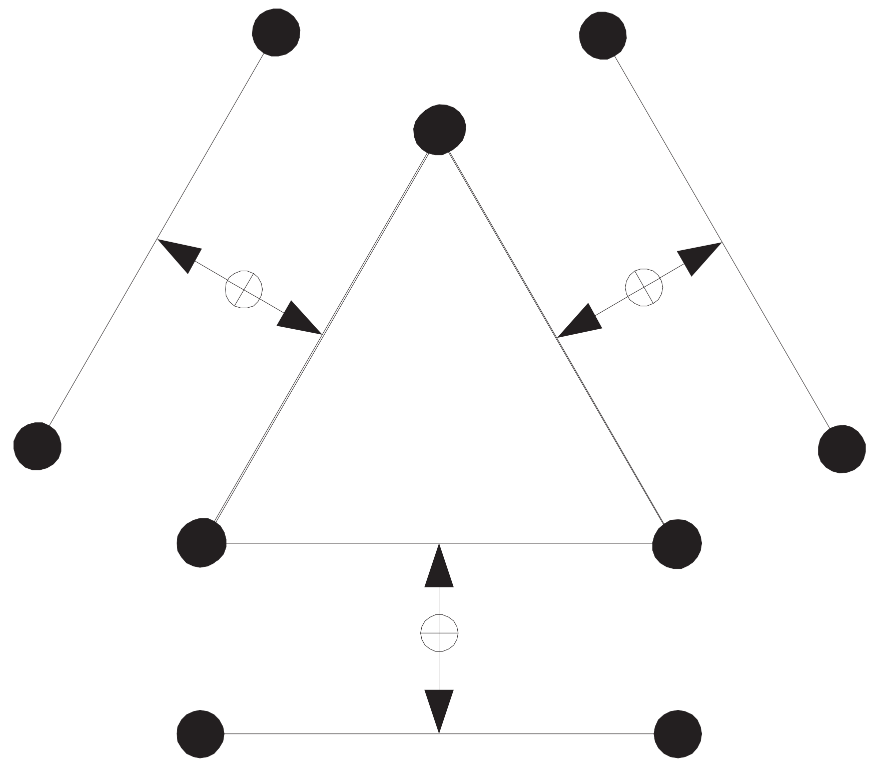


The Proof (continued)

- A Hamiltonian path that does not start or end at a node in a consistency gadget must travel it in one of two ways (drawn in green and red).
- Solid nodes are the only ones that connect to other gadgets.

The Proof (continued)

- Clauses will be turned into triangles.
- The choice and consistency gadgets make sure that each side of the triangle is traversed by the Hamiltonian path if and only if the corresponding literal is *false*.
- This implies that at least one literal has to be true if there is a Hamiltonian path.
  - If all three literals are false, then all edges of the triangle will be traversed, which is impossible (why?).



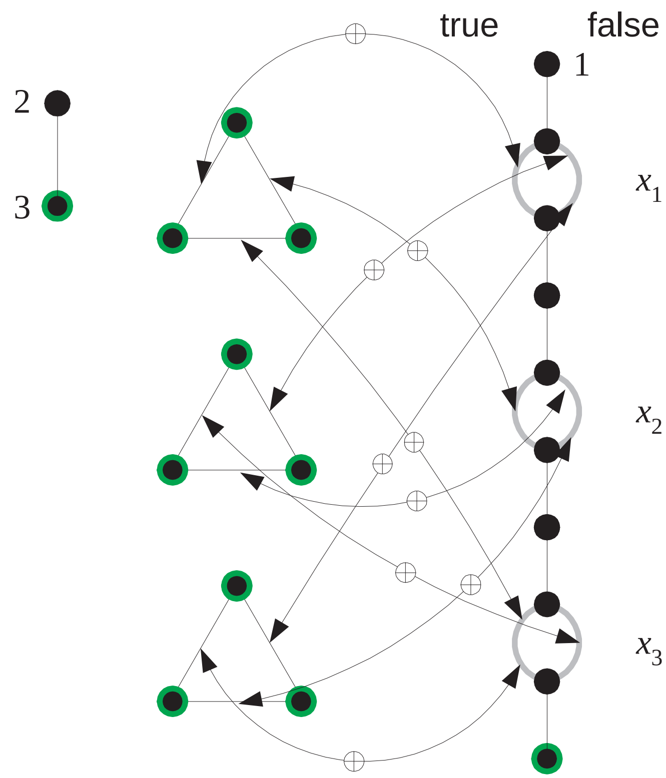
The Proof (continued)

- Graph $R(\phi)$ has n choice gadgets, one for each variable.
  - They are connected in series.
- Graph $R(\phi)$ has m triangles, one for each clause.
  - Each edge of the triangle corresponds to a literal in the clause.
  - An x_i edge is connected with a consistency gadget to the `true` edge of the choice gadget for x_i.
    - So the x_i edge is traversed if the `true` edge of the choice gadget is not.
  - A $\neg x_i$ edge is connected with a consistency gadget to the `false` edge of the choice gadget for x_i.
-

The Proof (continued)

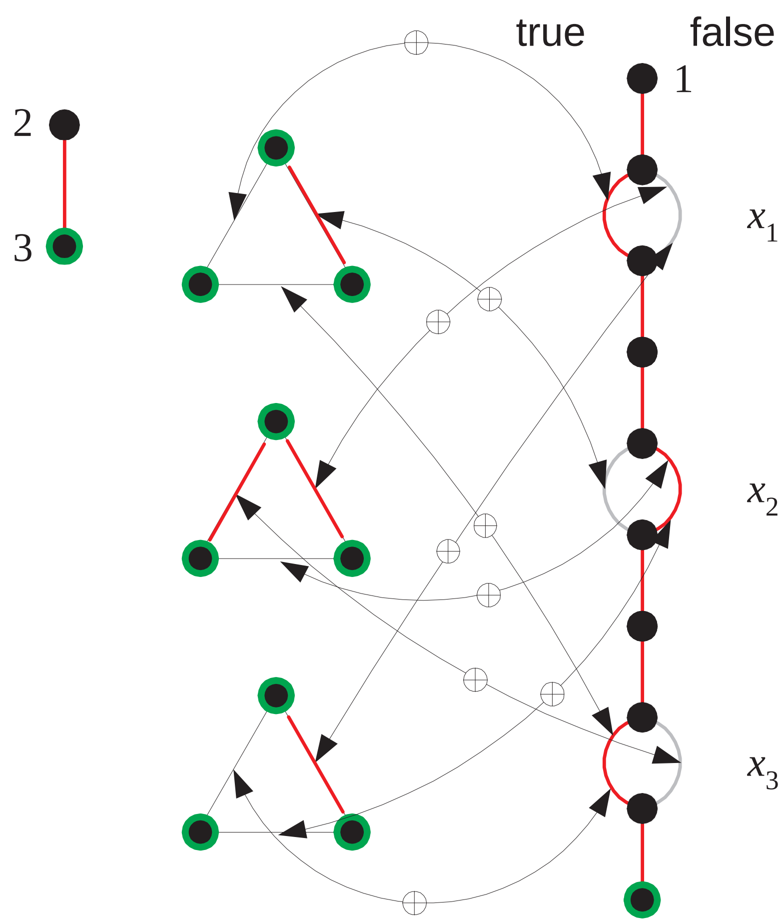
- All $3m$ nodes of the triangles plus the last node of the chain of choice gadgets and a new node 3 are connected by a complete graph (drawn in green).
- A single node 2 is connected to node 3.
- This finishes the construction of $R(\phi)$.

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$$



The Proof (continued)

- Suppose that a Hamiltonian path exists.
- It must start at node 1 and end at node 2.
- One of each variable's 2 parallel edges in the choice gadgets must be traversed.
- This defines a truth assignment T.
- Then the path traverses the triangles.


true
false
1
2
3
x_1
x_2
x_3

The Proof (continued)

- An edge of a triangle is traversed if and only if the corresponding literal is false.
- But not all sides of a triangle can be traversed.
- Hence $T \models \phi$.

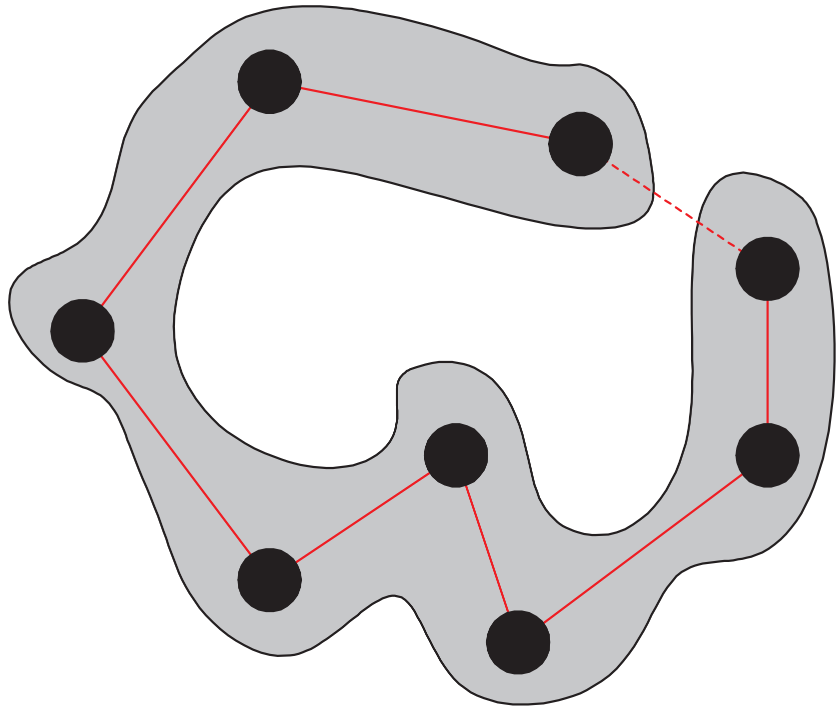
The Proof (concluded)

- Now suppose there is a truth assignment T that satisfies ϕ.
- We next find a Hamiltonian path of $R(\phi)$.
- The path starts at node 1.
- It traverses the edges of the choice gadgets whose corresponding literal is true under T.
- The rest of the graph is connected by a complete graph.
- We now traverse it (some of the green nodes on p. 341).

TSP (D) Is NP-Complete

Corollary 43 TSP (D) *is NP-complete.*

- Consider a graph G with n nodes.
- Create a weighted complete graph G' with the same nodes as from G follows.
- Set $d_{ij} = 1$ if $[\,i, j\,] \in G$ and $d_{ij} = 2$ if $[\,i, j\,] \notin G$.
- Set the budget $B = n + 1$.
- Suppose G has no Hamiltonian paths.
- Then every tour on G' must contain at least two edges with weight 2.
  - Otherwise, by removing up to one edge with weight 2, a Hamiltonian path for G obtains, a contradiction.



TSP (D) Is NP-Complete (concluded)

- The total cost is then at least $(n-2)+2\cdot 2 = n+2 > B$.
- On the other hand, suppose G has Hamiltonian paths.
- Then there is a tour on G' containing at most one edge with weight 2.
- The total cost is then at most $(n-1)+2 = n+1 = B$.
- We conclude that there is a tour of length B or less on G' if and only if G has a Hamiltonian path.

Graph Coloring

- k-COLORING: Can the nodes of a graph be colored with $\leq k$ colors such that no two adjacent nodes have the same color?
- 2-COLORING is in P (why?).
- But 3-COLORING is NP-complete (see next page).
- k-COLORING is NP-complete for $k \geq 3$ (why?).
- EXACT-k-COLORING asks if the nodes of a graph can be colored using exactly k colors.
- It remains NP-complete for $k \geq 3$ (why?).

3-COLORING Is NP-Complete[a]

- We will reduce NAESAT to 3-COLORING.
- We are given a set of clauses $C_1, C_2, \ldots, C_m$ each with 3 literals.
- The boolean variables are $x_1, x_2, \ldots, x_n$.
- We shall construct a graph G such that it can be colored with colors $\{0, 1, 2\}$ if and only if all the clauses can be NAE-satisfied.

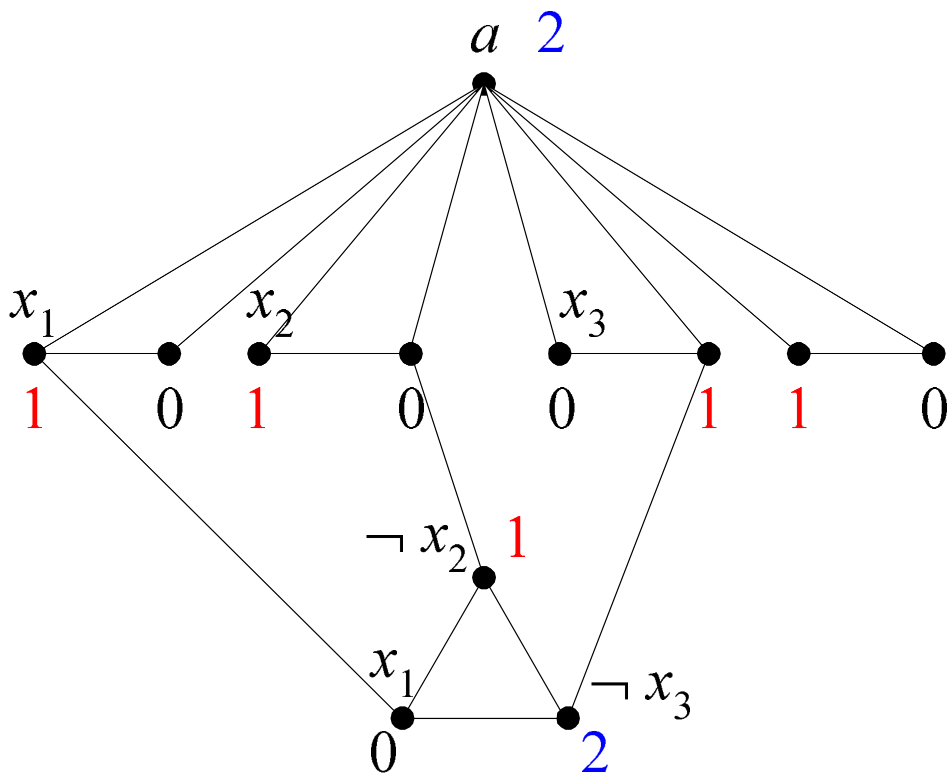
[a]Karp (1972).

The Proof (continued)

- Every variable x_i is involved in a triangle $[\,a, x_i, \neg x_i\,]$ with a common node a.
- Each clause $C_i = (c_{i1} \lor c_{i2} \lor c_{i3})$ is also represented by a triangle
$$[\,c_{i1}, c_{i2}, c_{i3}\,].$$
  - Node c_{ij} with the same label as one in some triangle $[\,a, x_k, \neg x_k\,]$ represent *distinct* nodes.
- There is an edge between c_{ij} and the node that represents the jth literal of C_i.
  - Alternative proof: there is an edge between $\neg c_{ij}$ and the node that represents the jth literal of C_i.[a]

[a] Contributed by Mr. Ren-Shuo Liu (D98922016) on October 27, 2009.

Construction for $\cdots \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge \cdots$



The Proof (continued)

Suppose the graph is 3-colorable.

- Assume without loss of generality that node a takes the color 2.
- A triangle must use up all 3 colors.
- As a result, one of x_i and $\neg x_i$ must take the color 0 and the other 1.

The Proof (continued)

- Treat 1 as `true` and 0 as `false`.[a]
  - We were dealing only with those triangles with the a node, not the clause triangles.
- The resulting truth assignment is clearly contradiction free.
- As each clause triangle contains one color 1 and one color 0, the clauses are NAE-satisfied.

[a] The opposite also works.

The Proof (continued)

Suppose the clauses are NAE-satisfiable.

- Color node a with color 2.
- Color the nodes representing literals by their truth values (color 0 for `false` and color 1 for `true`).
  - We were dealing only with those triangles with the a node, not the clause triangles.

The Proof (concluded)

- For each clause triangle:
  - Pick any two literals with opposite truth values.
  - Color the corresponding nodes with 0 if the literal is `true` and 1 if it is `false`.
  - Color the remaining node with color 2.
- The coloring is legitimate.
  - If literal w of a clause triangle has color 2, then its color will never be an issue.
  - If literal w of a clause triangle has color 1, then it must be connected up to literal w with color 0.
  - If literal w of a clause triangle has color 0, then it must be connected up to literal w with color 1.

Algorithms for 3-COLORING and the Chromatic Number $\chi(G)$

- Assume G is 3-colorable.
- There is an algorithm to find a 3-coloring in time $O(3^{n/3}) = 1.4422^n$.[a]
- It has been improved to $O(1.3289^n)$.[b]
- There is an algorithm to find $\chi(G)$ in time $O((4/3)^{n/3}) = 2.4422^n$.[c]
- It can be improved to $O((4/3 + 3^{4/3}/4)^n) = O(2.4150^n)$.[d]

[a] Lawler (1976).
[b] Beigel and Eppstein (2000).
[c] Lawler (1976).
[d] Eppstein (2003).

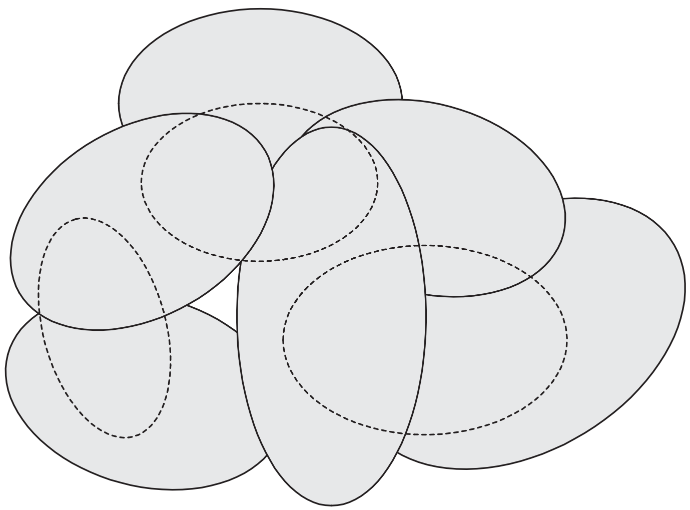
TRIPARTITE MATCHING

- We are given three sets B, G, and H, each containing n elements.
- Let $T \subseteq B \times G \times H$ be a ternary relation.
- TRIPARTITE MATCHING asks if there is a set of n triples in T, none of which has a component in common.
  - Each element in B is matched to a different element in G and different element in H.

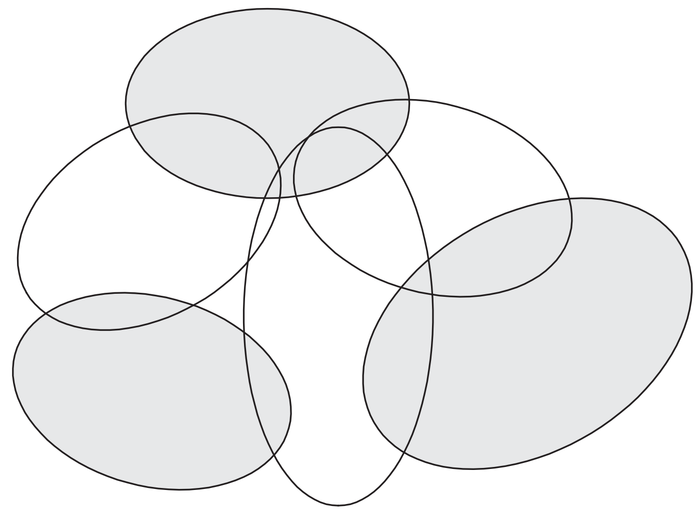
Theorem 44 (Karp (1972)) TRIPARTITE MATCHING *is NP-complete.*

Related Problems

- We are given a family $F = \{S_1, S_2, \ldots, S_n\}$ of subsets of a finite set U and a budget B.
- SET COVERING asks if there exists a set of B sets in F whose union is U.
- SET PACKING asks if there are B disjoint sets in F.
- Assume $|U| = 3m$ for some $m \in \mathbb{N}$ and $|S_i| = 3$ for all i.
- EXACT COVER BY 3-SETS asks if there are m sets in F that are disjoint and have U as their union.



SET COVERING



SET PACKING

Related Problems (concluded)

Corollary 45 SET COVERING, SET PACKING, *and* EXACT COVER BY 3-SETS *are all NP-complete.*

The KNAPSACK Problem

- There is a set of n items.
- Item i has value $v_i \in \mathbb{Z}^+$ and weight $w_i \in \mathbb{Z}^+$.
- We are given $K \in \mathbb{Z}^+$ and $W \in \mathbb{Z}^+$.
- KNAPSACK asks if there exists a subset $S \subseteq \{1, 2, \ldots, n\}$ such that $\sum_{i \in S} w_i \leq W$ and $\sum_{i \in S} v_i \geq K$.
  - We want to achieve the maximum satisfaction within the budget.

KNAPSACK Is NP-Complete[a]

- KNAPSACK $\in$ NP: Guess an S and verify the constraints.
- We assume $v_i = w_i$ for all i and $K = W$.
- KNAPSACK now asks if a subset of $\{v_1, v_2, \ldots, v_n\}$ adds up to exactly K.
  - Picture yourself as a radio DJ.
  - Or a person trying to control the calories intake.
- We shall reduce EXACT COVER BY 3-SETS to KNAPSACK.

[a] Karp (1972).

The Proof (continued)

- We are given a family $F = \{S_1, S_2, \ldots, S_n\}$ of size-3 subsets of $U = \{1, 2, \ldots, 3m\}$.
- EXACT COVER BY 3-SETS asks if there are m disjoint sets in F that cover the set U.
- Think of a set as a bit vector in $\{0, 1\}^{3m}$.
  - `001100010` means the set $\{3, 4, 8\}$, and `110010000` means the set $\{1, 2, 5\}$.
- Our goal is $\overbrace{\texttt{11} \cdots \texttt{1}}^{3m}$.

The Proof (continued)

- A bit vector can also be considered as a binary *number*.
- Set union resembles addition.
  - $\texttt{001100010} + \texttt{110010000} = \texttt{111110010}$, which denotes the set $\{1, 2, 3, 4, 5, 8\}$, as desired.
- Trouble occurs when there is *carry*.
  - $\texttt{001100010} + \texttt{001110000} = \texttt{010010010}$, which denotes the set $\{2, 5, 8\}$, not the desired $\{3, 4, 5, 8\}$.

The Proof (continued)

- Carry may also lead to a situation where we obtain our solution $\texttt{11}\cdots\texttt{1}$ with more than m sets in F.
  - $\texttt{001100010} + \texttt{001110000} + \texttt{101100000} + \texttt{000001101} = \texttt{111111111}$.
  - But this "solution" $\{1, 3, 4, 5, 6, 7, 8, 9\}$ does not correspond to an exact cover.
  - And it uses 4 sets instead of the required $m = 3$.[a]
- To fix this problem, we enlarge the base just enough so that there are no carries.
- Because there are n vectors in total, we change the base from 2 to $n + 1$.

[a]Thanks to a lively class discussion on November 20, 2002.

The Proof (continued)

- Set v_i to be the $(n+1)$-ary number corresponding to the bit vector encoding S_i.
- Now in base $n+1$, if there is a set S such that $\sum_{v_i \in S} v_i = \overbrace{11 \cdots 1}^{3m}$, then every bit position must be contributed by exactly one v_i and $|S| = m$.
- Finally, set

$$K = \sum_{j=0}^{3m-1} (n+1)^j = \overbrace{11 \cdots 1}^{3m} \quad \text{(base } n+1\text{)}.$$

The Proof (continued)

- Suppose F admits an exact cover, say $\{S_1, S_2, \ldots, S_m\}$.
- Then picking $S = \{v_1, v_2, \ldots, v_m\}$ clearly results in

$$v_1 + v_2 + \cdots + v_m = \overbrace{11 \cdots 1}^{3m}.$$

  - It is important to note that the meaning of addition (+) is independent of the base.[a]
  - It is just regular addition.
  - But a S_i may give rise to different v_i's under different bases.

---

[a]Contributed by Mr. Kuan-Yu Chen (`R92922047`) on November 3, 2004.

The Proof (concluded)

- On the other hand, suppose there exists an S such that $\sum_{v_i \in S} v_i = \overbrace{11 \cdots 1}^{3m}$ in base $n+1$.
- The no-carry property implies that $|S| = m$ and $\{S_i : v_i \in S\}$ is an exact cover.

An Example

- Let $m = 3$, $U = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, and

$$\begin{aligned} S_1 &= \{1, 3, 4\}, \\ S_2 &= \{2, 3, 4\}, \\ S_3 &= \{2, 5, 6\}, \\ S_4 &= \{6, 7, 8\}, \\ S_5 &= \{7, 8, 9\}. \end{aligned}$$

- Note that $n = 5$, as there are 5 S_i's.

An Example (concluded)

- Our reduction produces

$$
\begin{aligned}
K &= \sum_{j=0}^{3\times 3-1} 6^j = \overbrace{11\cdots 1}^{3\times 3} \quad \text{(base 6)} = 2015539,\\
v_1 &= \texttt{101100000} = 1734048,\\
v_2 &= \texttt{011100000} = 334368,\\
v_3 &= \texttt{010011000} = 281448,\\
v_4 &= \texttt{000001110} = 258,\\
v_5 &= \texttt{000000111} = 43.
\end{aligned}
$$

- Note $v_1 + v_3 + v_5 = K$.
- Indeed, $S_1 \cup S_3 \cup S_5 = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, an exact cover by 3-sets.

BIN PACKING

- We are given N positive integers $a_1, a_2, \ldots, a_N$, an integer C (the capacity), and an integer B (the number of bins).
- BIN PACKING asks if these numbers can be partitioned into B subsets, each of which has total sum at most C.
- Think of packing bags at the check-out counter.

Theorem 46 BIN PACKING *is NP-complete.*

INTEGER PROGRAMMING

- INTEGER PROGRAMMING asks whether a system of linear inequalities with integer coefficients has an integer solution.
  - LINEAR PROGRAMMING asks whether a system of linear inequalities with integer coefficients has a *rational* solution.

INTEGER PROGRAMMING Is NP-Complete[a]

- SET COVERING can be expressed by the inequalities $Ax \geq \vec{1}$, $\sum_{i=1}^{n} x_i \leq B$, $0 \leq x_i \leq 1$, where
  - x_i is one if and only if S_i is in the cover.
  - A is the matrix whose columns are the bit vectors of the sets $S_1, S_2, \ldots$.
  - $\vec{1}$ is the vector of 1s.
- This shows INTEGER PROGRAMMING is NP-hard.
- Many NP-complete problems can be expressed as an INTEGER PROGRAMMING problem.

[a] Papadimitriou (1981).

Christos Papadimitriou

Easier or Harder?[a]

- Adding restrictions on the allowable *problem instances* will not make a problem harder.
  - We are now solving a subset of problem instances.
  - The INDEPENDENT SET proof (p. 302) and the KNAPSACK proof (p. 361).
  - SAT to 2SAT (easier by p. 285).
  - CIRCUIT VALUE to MONOTONE CIRCUIT VALUE (equally hard by p. 257).

[a] Thanks to a lively class discussion on October 29, 2003.

Easier or Harder? (concluded)

- Adding restrictions on the allowable *solutions* may make a problem easier, as hard, or harder.
- It is problem dependent.
  - MIN CUT to BISECTION WIDTH (harder by p. 328).
  - LINEAR PROGRAMMING to INTEGER PROGRAMMING (harder by p. 371).
  - SAT to NAESAT (equally hard by p. 296) and MAX CUT to MAX BISECTION (equally hard by p. 326).
  - 3-COLORING to 2-COLORING (easier by p. 347).

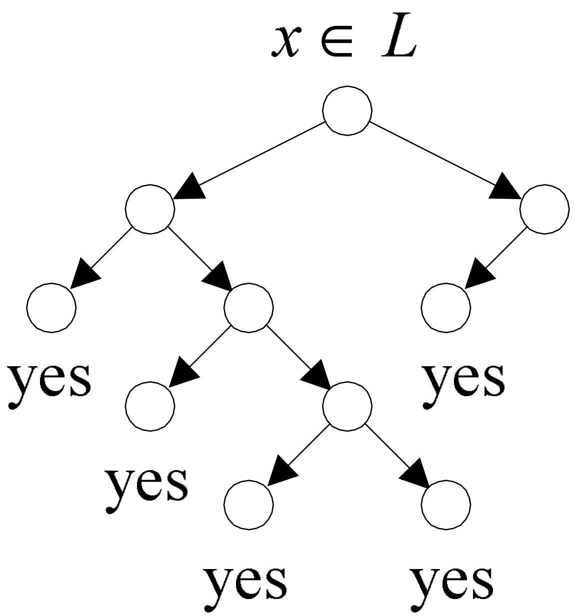
coNP and Function Problems

coNP

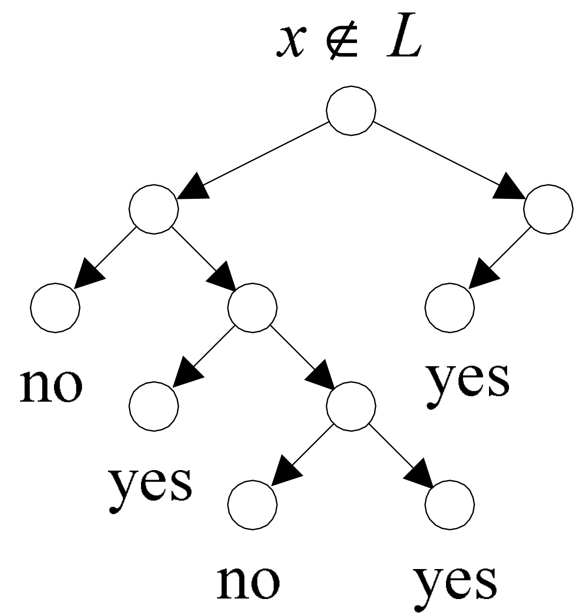
- By definition, coNP is the class of problems whose complement is in NP.
- NP is the class of problems that have succinct certificates (recall Proposition 34 on p. 267).
- coNP is therefore the class of problems that have succinct disqualifications:
  - A "no" instance of a problem in coNP possesses a short proof of its being a "no" instance.
  - Only "no" instances have such proofs.

coNP (continued)

- Suppose L is a coNP problem.
- There exists a polynomial-time nondeterministic algorithm M such that:
  - If $x \in L$, then $M(x) =$ "yes" for all computation paths.
  - If $x \notin L$, then $M(x) =$ "no" for some computation path.



$x \in L$
yes
yes
yes
yes
yes



$x \notin L$
no
yes
yes
no
yes

coNP (concluded)

- Clearly $\mathrm{P} \subseteq \mathrm{coNP}$.
- It is not known if

$$\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP}.$$

  - Contrast this with

$$\mathrm{R} = \mathrm{RE} \cap \mathrm{coRE}$$

    (see Proposition 10 on p. 128).

Some coNP Problems

- VALIDITY $\in$ coNP.
  - If ϕ is not valid, it can be disqualified very succinctly: a truth assignment that does not satisfy it.
- SAT COMPLEMENT $\in$ coNP.
  - The disqualification is a truth assignment that satisfies it.
- HAMILTONIAN PATH COMPLEMENT $\in$ coNP.
  - The disqualification is a Hamiltonian path.
- OPTIMAL TSP (D) $\in$ coNP.[a]
  - The disqualification is a tour with a length $< B$.

[a] Asked by Mr. Che-Wei Chang (R95922093) on September 27, 2006.

An Alternative Characterization of coNP

Proposition 47 *Let $L \subseteq \Sigma^*$ be a language. Then $L \in coNP$ if and only if there is a polynomially decidable and polynomially balanced relation R such that*

$$L = \{x : \forall y\, (x, y) \in R\}.$$

(As on p. 266, we assume $|\, y\,| \leq |\, x\,|^k$ for some k.)

- $\bar{L} = \{x : (x, y) \in \neg R \text{ for some } y\}$.
- Because $\neg R$ remains polynomially balanced, $\bar{L} \in$ NP by Proposition 34 (p. 267).
- Hence $L \in$ coNP by definition.

coNP Completeness

Proposition 48 *L is NP-complete if and only if its complement $\bar{L} = \Sigma^* - L$ is coNP-complete.*

Proof ($\Rightarrow$; the $\Leftarrow$ part is symmetric)

- Let $\bar{L}'$ be any coNP language.
- Hence $L' \in$ NP.
- Let R be the reduction from L' to L.
- So $x \in L'$ if and only if $R(x) \in L$.
- So $x \in \bar{L}'$ if and only if $R(x) \in \bar{L}$.
- R is a reduction from $\bar{L}'$ to $\bar{L}$.

Some coNP-Complete Problems

- SAT COMPLEMENT is coNP-complete.
  - SAT COMPLEMENT is the complement of SAT.
- VALIDITY is coNP-complete.
  - ϕ is valid if and only if $\neg\phi$ is not satisfiable.
  - The reduction from SAT COMPLEMENT to VALIDITY is hence easy.
- HAMILTONIAN PATH COMPLEMENT is coNP-complete.