NP-Complete Problems

Wir müssen wissen, wir werden wissen.
(We must know, we shall know.)
— David Hilbert (1900)

Two Notions

- Let $R \subseteq \Sigma^* \times \Sigma^*$ be a binary relation on strings.
- R is called **polynomially decidable** if

$$\{x; y : (x, y) \in R\}$$

  is in P.
- R is said to be **polynomially balanced** if $(x, y) \in R$ implies $|y| \leq |\, x \,|^k$ for some $k \geq 1$.

An Alternative Characterization of NP

Proposition 30 (Edmonds (1965)) *Let $L \subseteq \Sigma^*$ be a language. Then $L \in NP$ if and only if there is a polynomially decidable and polynomially balanced relation R such that*

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

- Suppose such an R exists.
- L can be decided by this NTM:
  - On input x, the NTM guesses a y of length $\leq |\, x\,|^k$ and tests if $(x, y) \in R$ in polynomial time.
  - It returns "yes" if the test is positive.

The Proof (concluded)

- Now suppose $L \in \text{NP}$.
- NTM N decides L in time $|\,x\,|^k$.
- Define R as follows: $(x, y) \in R$ if and only if y is the encoding of an accepting computation of N on input x.
- Clearly R is polynomially balanced because N is polynomially bounded.
- R is polynomially decidable because it can be efficiently verified by checking with N's transition function.
- Finally $L = \{x : (x, y) \in R \text{ for some } y\}$ because N decides L.

Comments

- Any "yes" instance x of an NP problem has at least one **succinct certificate** or **polynomial witness** y.
- "No" instances have none.
- Certificates are short and easy to verify.
  - An alleged satisfying truth assignment for SAT; an alleged Hamiltonian path for HAMILTONIAN PATH.
- Certificates may be hard to generate (otherwise, NP equals P), but verification must be easy.
- NP is the class of *easy-to-verify* (in P) problems.

You Have an NP-Complete Problem (for Your Thesis)

- From Propositions 24 (p. 221) and Proposition 25 (p. 224), it is the least likely to be in P.
- Your options are:
  - Approximations.
  - Special cases.
  - Average performance.
  - Randomized algorithms.
  - Exponential-time algorithms that work well in practice.
  - "Heuristics" (and pray).

3SAT

- k-SAT, where $k \in \mathbb{Z}^+$, is the special case of SAT.
- The formula is in CNF and all clauses have *exactly* k literals (repetition of literals is allowed).
- For example,

$$(x_1 \lor x_2 \lor \neg x_3) \land (x_1 \lor x_1 \lor \neg x_2) \land (x_1 \lor \neg x_2 \lor \neg x_3).$$

3SAT Is NP-Complete

- Recall Cook's Theorem (p. 242) and the reduction of CIRCUIT SAT to SAT (p. 210).
- The resulting CNF has at most 3 literals for each clause.
  - This shows that 3SAT where each clause has at most 3 literals is NP-complete.
- Finally, duplicate one literal once or twice to make it a 3SAT formula.
- Note: The overall reduction remains parsimonious.

The Satisfiability of Random 3SAT Expressions

- Consider a random 3SAT expressions ϕ with n variables and cn clauses.
- Each clause is chosen independently and uniformly from the set of all possible clauses.
- Intuitively, the larger the c, the less likely ϕ is satisfiable as more constraints are added.
- Indeed, there is a c_n such that for $c < c_n(1 - \epsilon)$, ϕ is satisfiable almost surely, and for $c > c_n(1 + \epsilon)$, ϕ is unsatisfiable almost surely.[a]

[a] Friedgut and Bourgain (1999). As of 2006, $3.52 < c_n < 4.596$.

Another Variant of 3SAT

Proposition 31 3SAT *is NP-complete for expressions in which each variable is restricted to appear at most three times, and each literal at most twice. (*3SAT *here requires only that each clause has* at most *3 literals.)*

- Consider a general 3SAT expression in which x appears k times.
- Replace the first occurrence of x by x_1, the second by x_2, and so on, where $x_1, x_2, \ldots, x_k$ are k *new* variables.

The Proof (concluded)

- Add $(\neg x_1 \lor x_2) \land (\neg x_2 \lor x_3) \land \cdots \land (\neg x_k \lor x_1)$ to the expression.
  - This is logically equivalent to $x_1 \Rightarrow x_2 \Rightarrow \cdots \Rightarrow x_k \Rightarrow x_1$.
  - Note that each clause above has fewer than 3 literals.
- The resulting equivalent expression satisfies the condition for x.

An Example

- Suppose we are given the following 3SAT expression

$$\cdots (\neg x \vee w \vee g) \wedge \cdots \wedge (x \vee y \vee z) \cdots .$$

- The transformed expression is

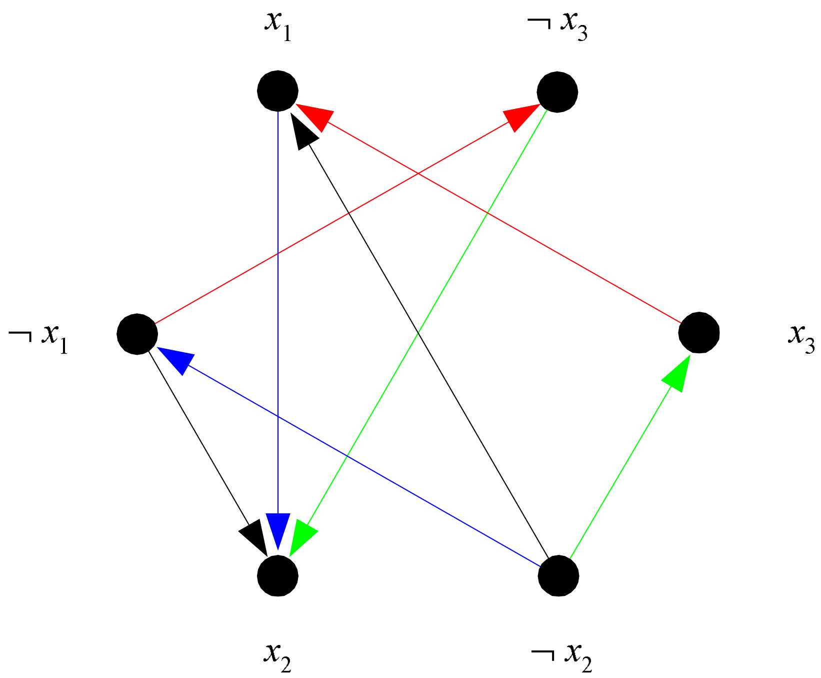
$$\cdots (\neg x_1 \vee w \vee g) \wedge \cdots \wedge (x_2 \vee y \vee z) \cdots (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_1).$$

  - Variable x_1 appears thrice.
  - Literal x_1 appears once.
  - Literal $\neg x_1$ appears twice.

2SAT and Graphs

- Let ϕ be an instance of 2SAT: Each clause has 2 literals.
- Define graph $G(\phi)$ as follows:
  - The nodes are the variables and their negations.
  - Add edges $(\neg\alpha, \beta)$ and $(\neg\beta, \alpha)$ to $G(\phi)$ if $\alpha \vee \beta$ is a clause in ϕ.
    * For example, if $x \vee \neg y \in \phi$, add $(\neg x, \neg y)$ and (y, x).
    * *Two* edges are added for each clause.
- Think of the edges as $\neg\alpha \Rightarrow \beta$ and $\neg\beta \Rightarrow \alpha$.
- b is reachable from a iff $\neg a$ is reachable from $\neg b$.
- Paths in $G(\phi)$ are valid implications.

Illustration: Directed Graph for $(x_1 \lor x_2) \land (x_1 \lor \neg x_3) \land (\neg x_1 \lor x_2) \land (x_2 \lor x_3)$



Properties of $G(\phi)$

Theorem 32 *ϕ is unsatisfiable if and only if there is a variable x such that there are paths from x to $\neg x$ and from $\neg x$ to x in $G(\phi)$.*

2SAT Is in NL $\subseteq$ P

- NL is a subset of P (p. 181).
- By Eq. (3) on p. 191, coNL equals NL.
- We need to show only that recognizing unsatisfiable expressions is in NL.
- In nondeterministic logarithmic space, we can test the conditions of Theorem 32 (p. 263) by guessing a variable x and testing if $\neg x$ is reachable from x *and* if $\neg x$ can reach x.
  - See the algorithm for REACHABILITY (p. 94).

Generalized 2SAT: MAX2SAT

- Consider a 2SAT expression.
- Let $K \in \mathbb{N}$.
- MAX2SAT is the problem of whether there is a truth assignment that satisfies at least K of the clauses.
- MAX2SAT becomes 2SAT when K equals the number of clauses.
- MAX2SAT is an optimization problem.
- MAX2SAT $\in$ NP: Guess a truth assignment and verify the count.

MAX2SAT Is NP-Complete[a]

- Consider the following 10 clauses:

$$(x) \wedge (y) \wedge (z) \wedge (w)$$
$$(\neg x \vee \neg y) \wedge (\neg y \vee \neg z) \wedge (\neg z \vee \neg x)$$
$$(x \vee \neg w) \wedge (y \vee \neg w) \wedge (z \vee \neg w)$$

- Let the 2SAT formula $r(x, y, z, w)$ represent the conjunction of these clauses.
- How many clauses can we satisfy?
- The clauses are symmetric with respect to x, y, and z.

[a]Garey, Johnson, and Stockmeyer (1976).

The Proof (continued)

All of x, y, z are true: By setting w to true, we satisfy $4 + 0 + 3 = 7$ clauses, whereas by setting w to false, we satisfy only $3 + 0 + 3 = 6$ clauses.

Two of x, y, z are true: By setting w to true, we satisfy $3 + 2 + 2 = 7$ clauses, whereas by setting w to false, we satisfy $2 + 2 + 3 = 7$ clauses.

The Proof (continued)

One of x, y, z is true: By setting w to false, we satisfy $1 + 3 + 3 = 7$ clauses, whereas by setting w to true, we satisfy only $2 + 3 + 1 = 6$ clauses.

None of x, y, z is true: By setting w to false, we satisfy $0 + 3 + 3 = 6$ clauses, whereas by setting w to true, we satisfy only $1 + 3 + 0 = 4$ clauses.

The Proof (continued)

- Any truth assignment that satisfies $x \vee y \vee z$ can be extended to satisfy 7 of the 10 clauses and no more.
- Any other truth assignment can be extended to satisfy only 6 of them.
- The reduction from 3SAT ϕ to MAX2SAT $R(\phi)$:
  - For each clause $C_i = (\alpha \vee \beta \vee \gamma)$ of ϕ, add **group** $r(\alpha, \beta, \gamma, w_i)$ to $R(\phi)$.
  - If ϕ has m clauses, then $R(\phi)$ has $10m$ clauses.
- Set $K = 7m$.

The Proof (concluded)

- We now show that K clauses of $R(\phi)$ can be satisfied if and only if ϕ is satisfiable.
- Suppose $7m$ clauses of $R(\phi)$ can be satisfied.
  - 7 clauses must be satisfied in each group because each group can have at most 7 clauses satisfied.
  - Hence all clauses of ϕ must be satisfied.
- Suppose all clauses of ϕ are satisfied.
  - Each group can set its w_i appropriately to have 7 clauses satisfied.

NAESAT

- The NAESAT (for "not-all-equal" SAT) is like 3SAT.
- But we require additionally that there be a satisfying truth assignment under which no clauses have the three literals equal in truth value.
  - Each clause must have one literal assigned true and one literal assigned false.

NAESAT Is NP-Complete[a]

- Recall the reduction of CIRCUIT SAT to SAT on p. 210.
- It produced a CNF ϕ in which each clause has at most 3 literals.
- Add the same variable z to all clauses with fewer than 3 literals to make it a 3SAT formula.
- Goal: The new formula $\phi(z)$ is NAE-satisfiable if and only if the original circuit is satisfiable.

[a]Karp (1972).

The Proof (continued)

- Suppose T NAE-satisfies $\phi(z)$.
  - $\bar{T}$ also NAE-satisfies $\phi(z)$.
  - Under T or $\bar{T}$, variable z takes the value false.
  - This truth assignment must still satisfy all clauses of ϕ.
  - So it satisfies the original circuit.

The Proof (concluded)

- Suppose there is a truth assignment that satisfies the circuit.
  - Then there is a truth assignment T that satisfies every clause of ϕ.
  - Extend T by adding $T(z) = \texttt{false}$ to obtain T'.
  - T' satisfies $\phi(z)$.
  - So in no clauses are all three literals false under T'.
  - Under T', in no clauses are all three literals true.
    * Review the detailed construction on p. 211 and p. 212.

Undirected Graphs

- An **undirected graph** $G = (V, E)$ has a finite set of nodes, V, and a set of *undirected* edges, E.
- It is like a directed graph except that the edges have no directions and there are no self-loops.
- We use $[\, i, j \,]$ to denote the fact that there is an edge between node i and node j.

Independent Sets

- Let $G = (V, E)$ be an undirected graph.
- $I \subseteq V$.
- I is **independent** if whenever $i, j \in I$, there is no edge between i and j.
- The INDEPENDENT SET problem: Given an undirected graph and a goal K, is there an independent set of size K?
  - Many applications.

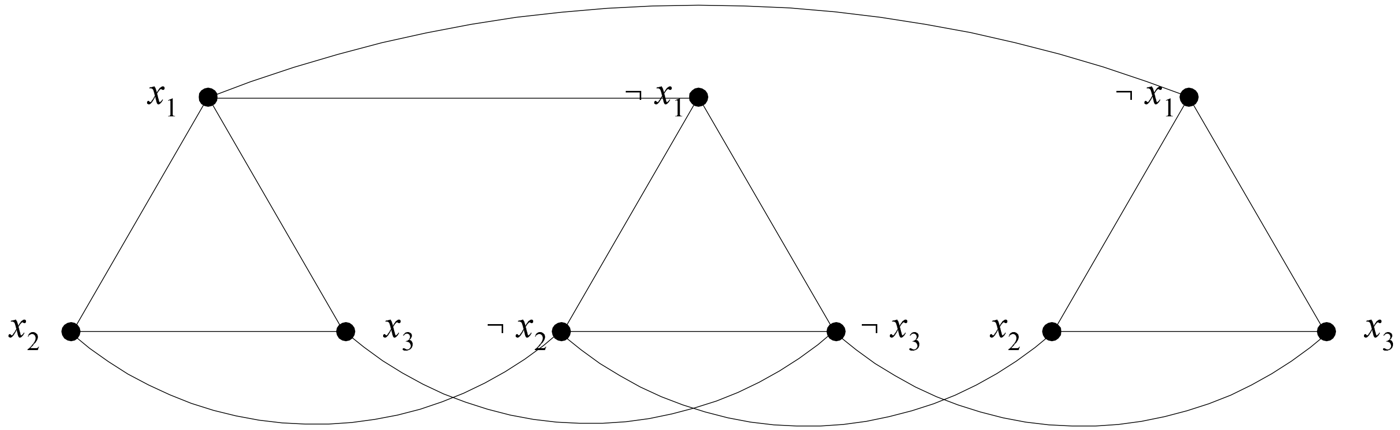
INDEPENDENT SET Is NP-Complete

- This problem is in NP: Guess a set of nodes and verify that it is independent and meets the count.
- If a graph contains a triangle, any independent set can contain at most one node of the triangle.
- We consider graphs whose nodes can be partitioned in m disjoint triangles.
  - If the special case is hard, the original problem must be at least as hard.

Reduction from 3SAT to INDEPENDENT SET

- Let ϕ be an instance of 3SAT with m clauses.
- We will construct graph G (with constraints as said) with $K = m$ such that ϕ is satisfiable if and only if G has an independent set of size K.
- There is a triangle for each clause with the literals as the nodes.
- Add additional edges between x and $\neg x$ for every variable x.

A Sample Construction



$$(x_1 \lor x_2 \lor x_3) \land (\neg x_1 \lor \neg x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3).$$

The Proof (continued)

- Suppose G has an independent set I of size $K = m$.
  - An independent set can contain at most m nodes, one from each triangle.
  - An independent set of size m exists if and only if it contains exactly one node from each triangle.
  - Truth assignment T assigns true to those literals in I.
  - T is consistent because contradictory literals are connected by an edge, hence not both in I.
  - T satisfies ϕ because it has a node from every triangle, thus satisfying every clause.

The Proof (concluded)

- Suppose a satisfying truth assignment T exists for ϕ.
  - Collect one node from each triangle whose literal is true under T.
  - The choice is arbitrary if there is more than one true literal.
  - This set of m nodes must be independent by construction.
    - Literals x and $\neg x$ cannot be both assigned true.

Other INDEPENDENT SET-Related NP-Complete Problems

Corollary 33 4-DEGREE INDEPENDENT SET *is NP-complete.*

Theorem 34 INDEPENDENT SET *is NP-complete for planar graphs.*

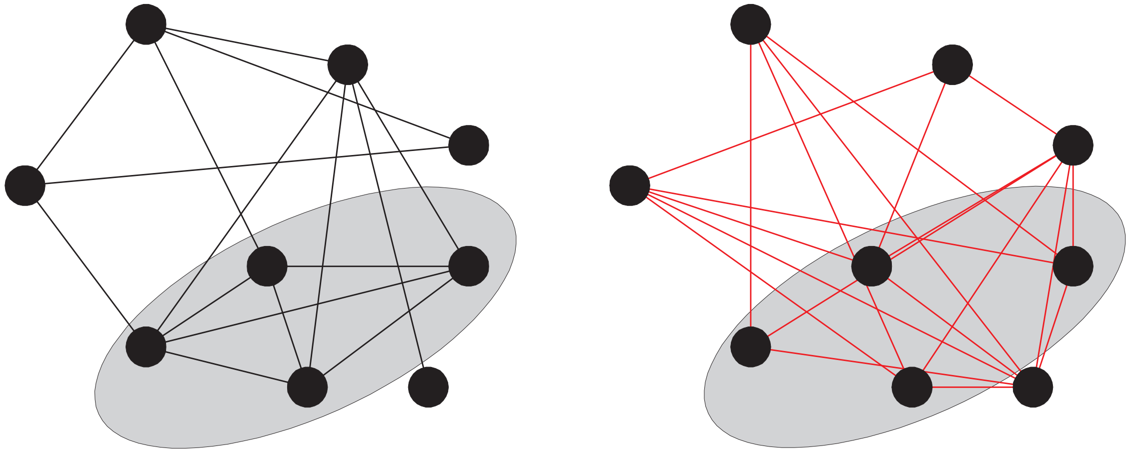
CLIQUE

- We are given an undirected graph G and a goal K.
- CLIQUE asks if there is a set C with K nodes such that whenever $i, j \in C$, there is an edge between i and j.

CLIQUE Is NP-Complete

Corollary 35 CLIQUE *is NP-complete.*

- Let $\bar{G}$ be the **complement** of G, where $[x, y] \in \bar{G}$ if and only if $[x, y] \notin G$.
- I is an independent set in $G \Leftrightarrow I$ is a clique in $\bar{G}$.



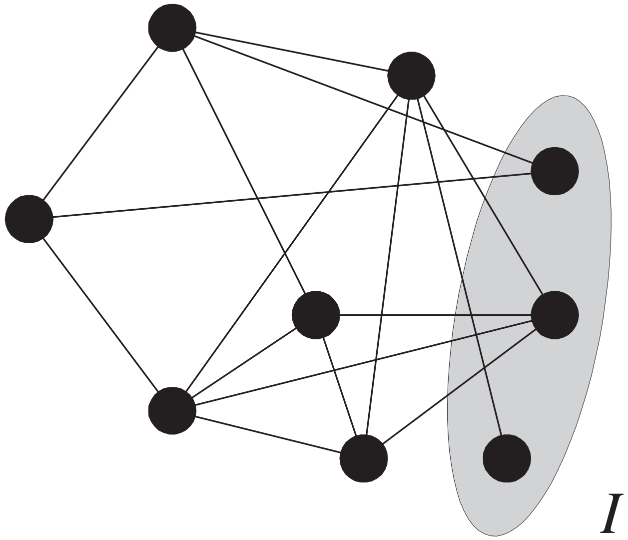
NODE COVER

- We are given an undirected graph G and a goal K.
- NODE COVER asks if there is a set C with K or fewer nodes such that each edge of G has at least one of its endpoints in C.

NODE COVER Is NP-Complete

Corollary 36 NODE COVER *is NP-complete.*

- I is an independent set of $G = (V, E)$ if and only if $V - I$ is a node cover of G.



MIN CUT and MAX CUT

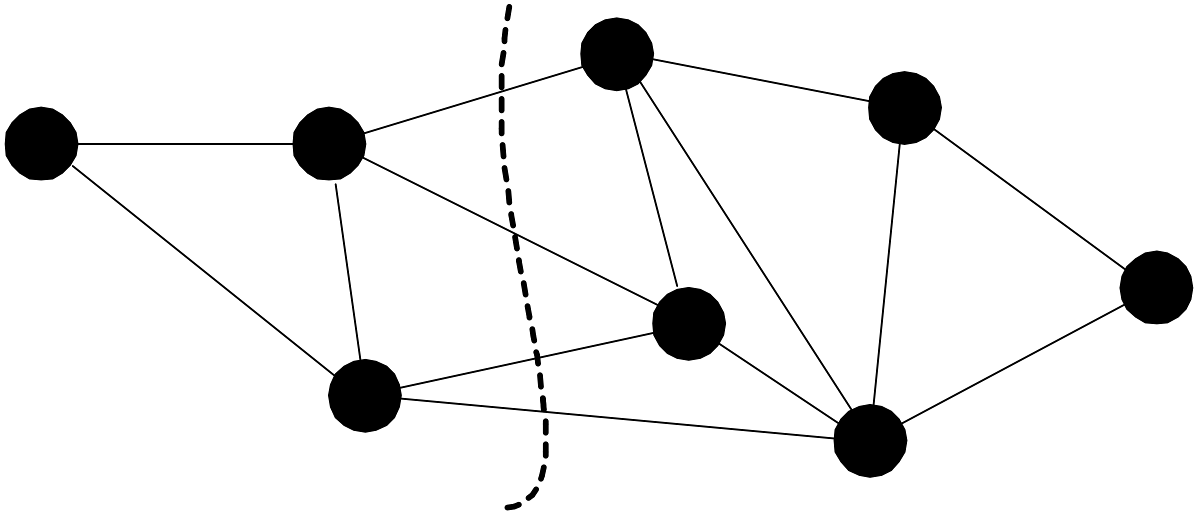
- A **cut** in an undirected graph $G = (V, E)$ is a partition of the nodes into two nonempty sets S and $V - S$.
- The size of a cut $(S, V - S)$ is the number of edges between S and $V - S$.
- MIN CUT $\in$ P by the maxflow algorithm.
- MAX CUT asks if there is a cut of size at least K.
  - K is part of the input.

MIN CUT and MAX CUT (concluded)

- MAX CUT has applications in VLSI layout.
  - The minimum area of a VLSI layout of a graph is not less than the square of its maximum cut size.[a]

[a] Raspaud, Sýkora, and Vrťo (1995).

A Cut



MAX CUT Is NP-Complete[a]

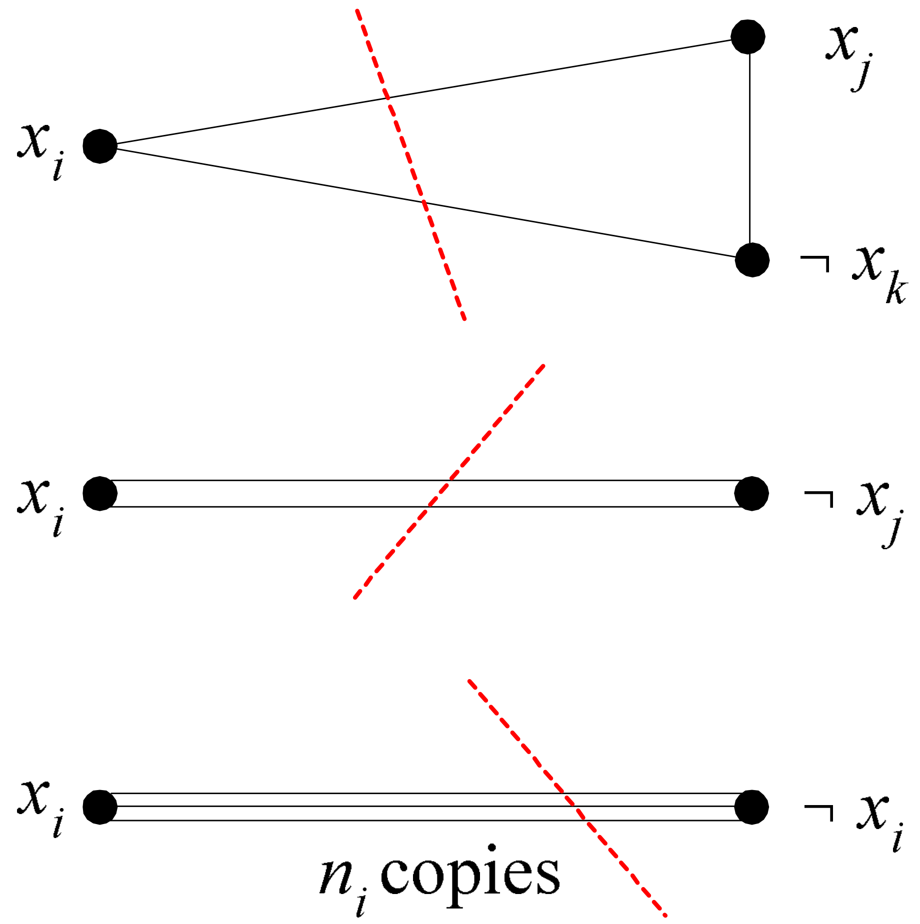
- We will reduce NAESAT to MAX CUT.
- Given an instance ϕ of 3SAT with m clauses, we shall construct a graph $G = (V, E)$ and a goal K such that:
  - There is a cut of size at least K if and only if ϕ is NAE-satisfiable.
- Our graph will have multiple edges between two nodes.
  - Each such edge contributes one to the cut if its nodes are separated.

[a]Garey, Johnson, and Stockmeyer (1976).

The Proof

- Suppose ϕ's m clauses are $C_1, C_2, \ldots, C_m$.
- The boolean variables are $x_1, x_2, \ldots, x_n$.
- G has $2n$ nodes: $x_1, x_2, \ldots, x_n, \neg x_1, \neg x_2, \ldots, \neg x_n$.
- Each clause with 3 distinct literals makes a triangle in G.
- For each clause with two identical literals, there are two parallel edges between the two distinct literals.
- No need to consider clauses with one literal (why?).
- For each variable x_i, add n_i copies of edge $[x_i, \neg x_i]$, where n_i is the number of occurrences of x_i and $\neg x_i$ in ϕ.[a]

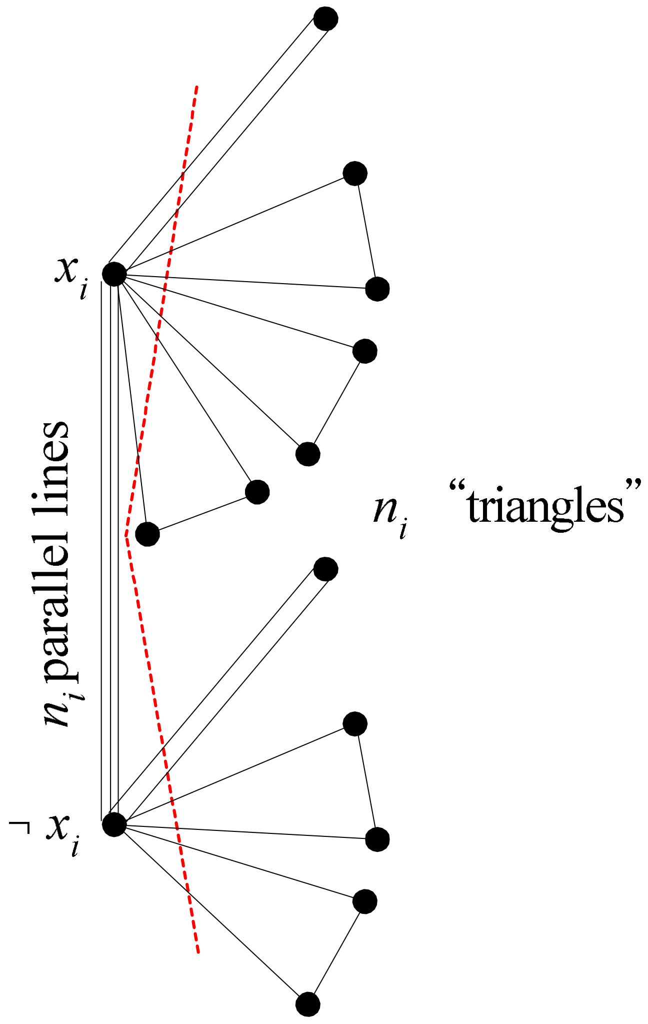
[a]Regardless of whether both x_i and $\neg x_i$ occur in ϕ.



x_i
x_j
$\neg\, x_k$
x_i
$\neg\, x_j$
x_i
$\neg\, x_i$
n_i copies

The Proof (continued)

- Set $K = 5m$.
- Suppose there is a cut $(S, V - S)$ of size $5m$ or more.
- A clause (a triangle or two parallel edges) contributes at most 2 to a cut no matter how you split it.
- Suppose both x_i and $\neg x_i$ are on the same side of the cut.
- Then they *together* contribute at most $2n_i$ edges to the cut as they appear in at most n_i different clauses.



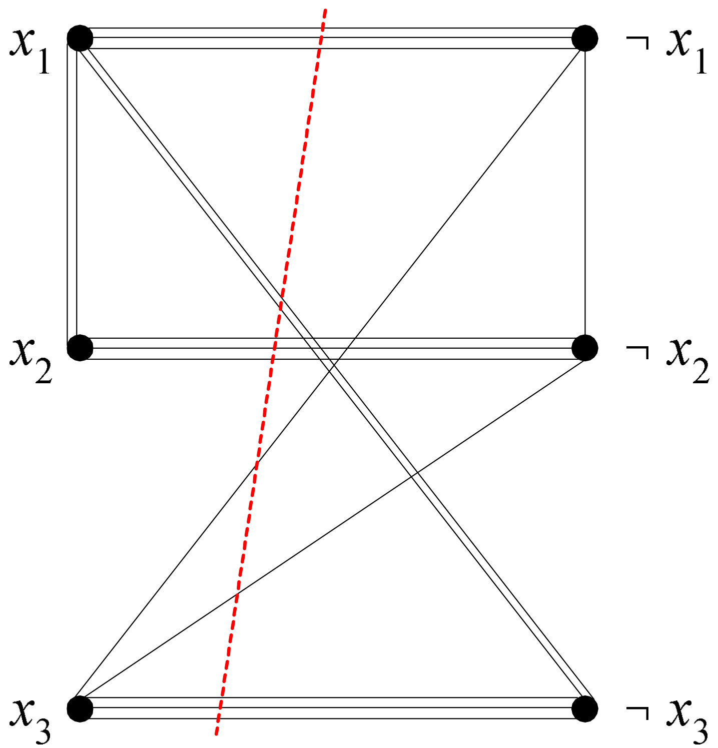
x_i
n_i parallel lines
$\neg x_i$
n_i "triangles"

The Proof (continued)

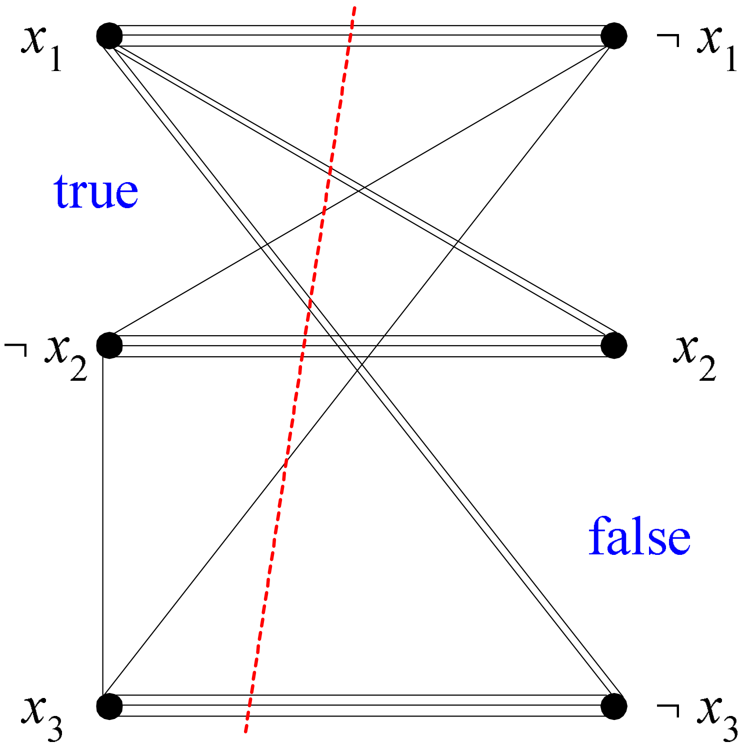
- Changing the side of a literal contributing at most n_i to the cut does not decrease the size of the cut.
- Hence we assume variables are separated from their negations.
- The total number of edges in the cut that join opposite literals is $\sum_i n_i = 3m$.
  - The total number of literals is $3m$.

The Proof (concluded)

- The *remaining* $2m$ edges in the cut must come from the m triangles or parallel edges that correspond to the clauses.
- As each can contribute at most 2 to the cut, all are split.
- A split clause means at least one of its literals is true and at least one false.
- The other direction is left as an exercise.



- $(x_1 \vee x_2 \vee x_2) \wedge (x_1 \vee \neg x_3 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$.
- The cut size is $13 < 5 \times 3 = 15$.



- $(x_1 \lor x_2 \lor x_2) \land (x_1 \lor \neg x_3 \lor \neg x_3) \land (\neg x_1 \lor \neg x_2 \lor x_3)$.
- The cut size is now 15.

A Remark

- 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]

---

[a]Contributed by Mr. Tai-Dai Chou (J93922005) on June 2, 2005.
[b]Contributed by Mr. Chien-Lin Chen (J94922015) on June 8, 2006.

MAX BISECTION

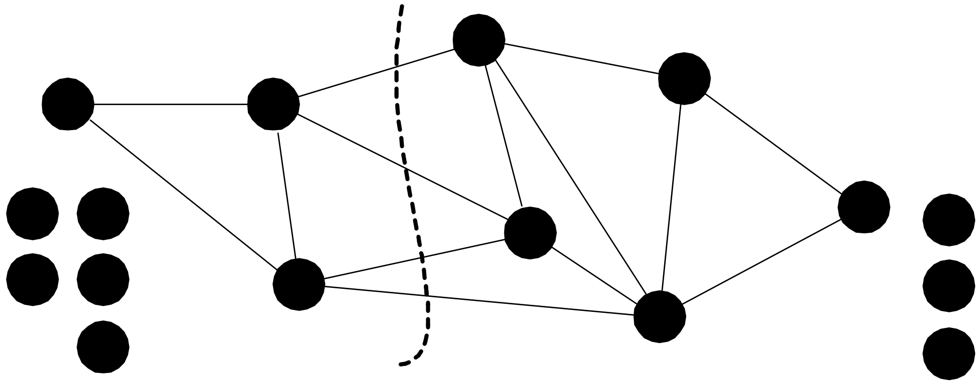
- MAX CUT becomes MAX BISECTION if we require that $|S| = |V - S|$.
- It has many applications, especially in VLSI layout.

MAX BISECTION Is NP-Complete

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

The Proof (concluded)

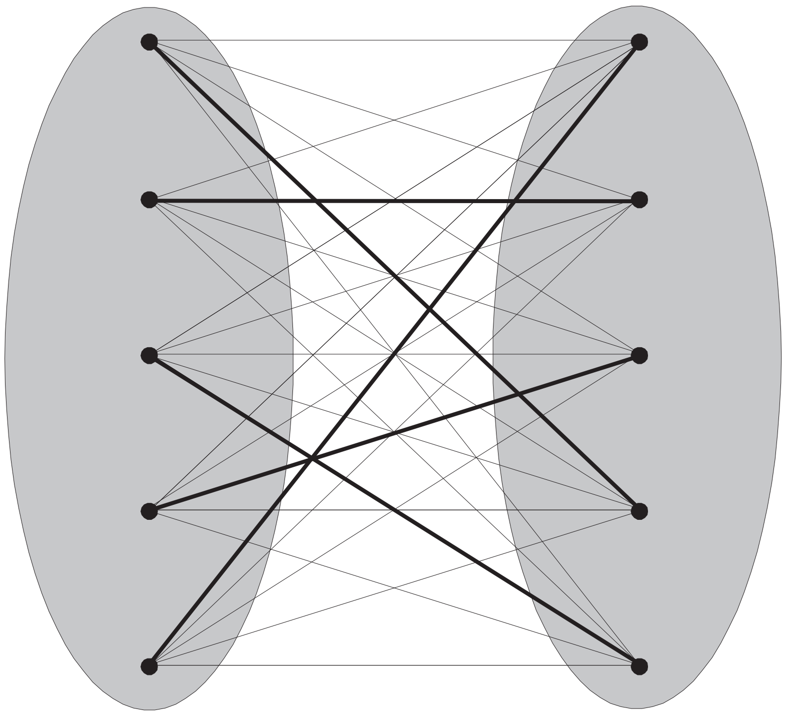
- Every cut $(S, V - S)$ of $G = (V, E)$ can be made into a bisection by appropriately allocating the new nodes between S and $V - S$.
- Hence each cut of G can be made a cut of G' of the same size, and vice versa.



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

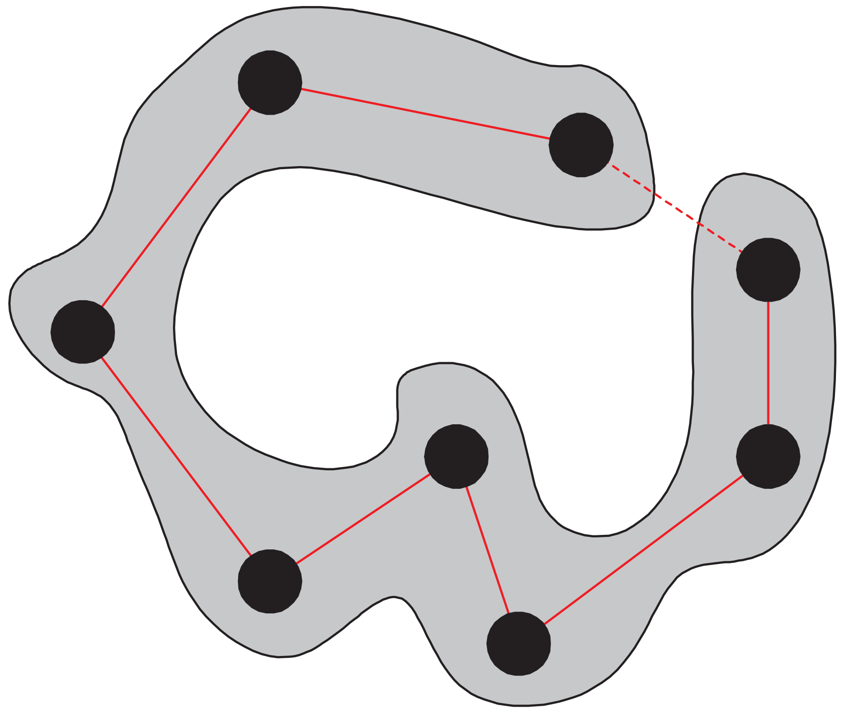
---

[a] Karp (1972).

TSP (D) Is NP-Complete

Corollary 38 TSP (D) *is NP-complete.*

- Consider a graph G with n nodes.
- Define $d_{ij} = 1$ 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 the new graph must contain at least two edges with weight 2.
  - Otherwise, by removing up to one edge with weight 2, one obtains a Hamiltonian path, 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 the new graph 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 if and only if G has a Hamiltonian path.