Cook's Theorem: the First NP-Complete Problem

Theorem 37 (Cook (1971)) SAT *is NP-complete.*

- SAT $\in$ NP (p. 113).
- CIRCUIT SAT reduces to SAT (p. 284).
- Now we only need to show that all languages in NP can be reduced to CIRCUIT SAT.[a]

[a]As a bonus, this also shows CIRCUIT SAT is NP-complete.

The Proof (continued)

- Let single-string NTM M decide $L \in \mathrm{NP}$ in time n^k.
- Assume M has exactly *two* nondeterministic choices at each step: choices 0 and 1.
- For each input x, we construct circuit $R(x)$ such that $x \in L$ if and only if $R(x)$ is satisfiable.
- Equivalently, for each input x, $M(x) =$ "yes" for some computation path if and only if $R(x)$ is satisfiable.
- How to come up with a polynomial-sized $R(x)$ when there are exponentially many computation paths?

The Proof (continued)

- A straightforward proof is to construct a variable-free circuit $R_i(x)$ for the ith computation path.[a]
- Then add a small circuit to output 1 if and only if there is an $R_i(x)$ that outputs a "yes."
- Clearly, the resulting circuit outputs 1 if and only if M accepts x.
- But, it is too large because there are exponentially many computation paths.

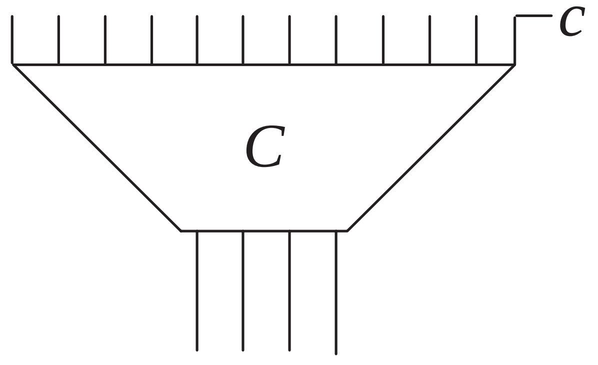
[a]The circuit for Theorem 34 (p. 305) will do.

The Proof (continued)

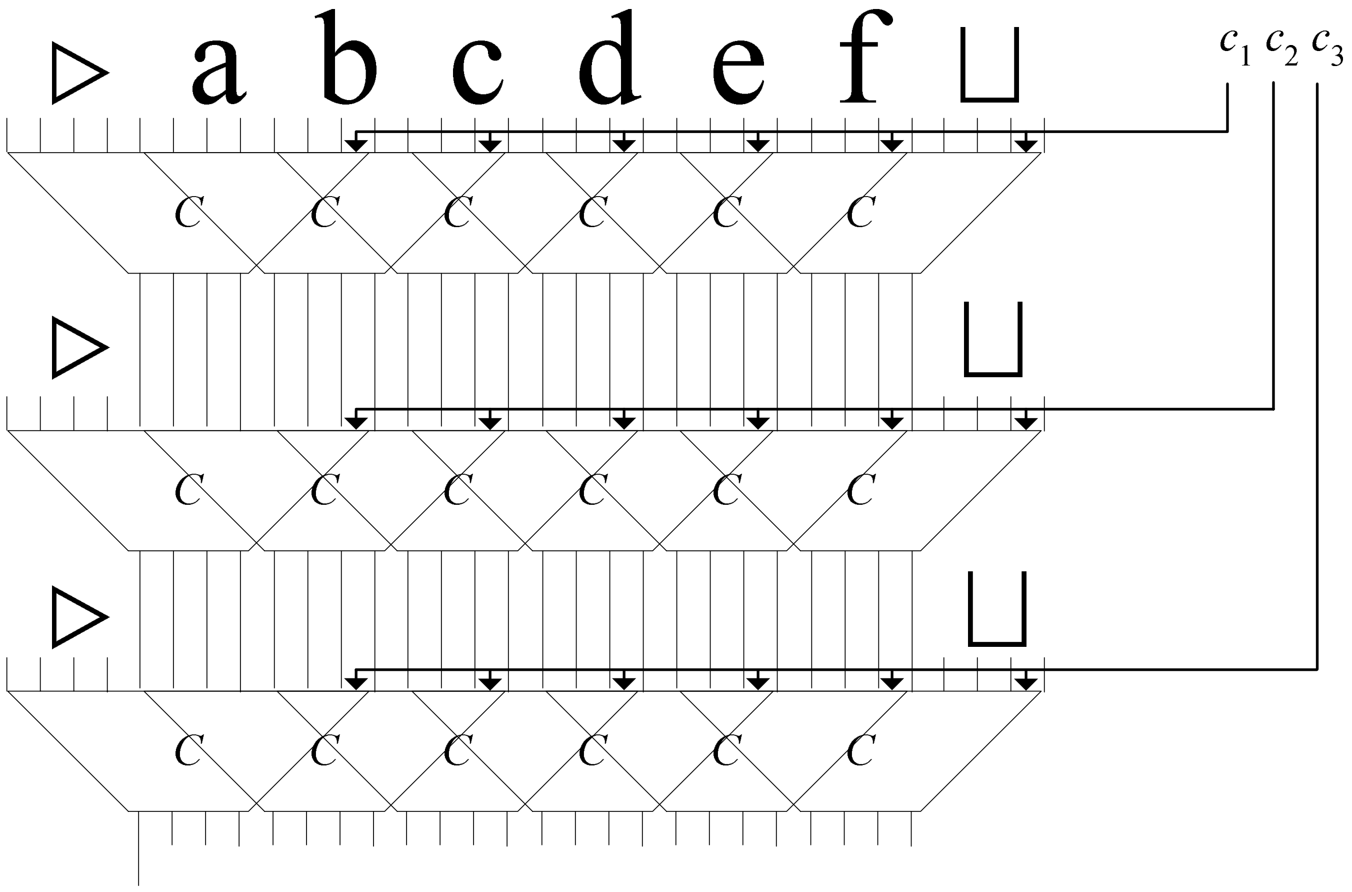
- A sequence of nondeterministic choices is a bit string

$$B = (c_1, c_2, \ldots, c_{|\,x\,|^k-1}) \in \{0,1\}^{|\,x\,|^k-1}.$$

- Once B is given, the computation is *deterministic*.
- Each choice of B results in a deterministic polynomial-time computation.
- Each circuit C at time i has an extra binary input c corresponding to the nondeterministic choice: $C(T_{i-1,j-1}, T_{i-1,j}, T_{i-1,j+1}, c) = T_{ij}$.


c
C

The Computation Tableau for NTMs and $R(x)$

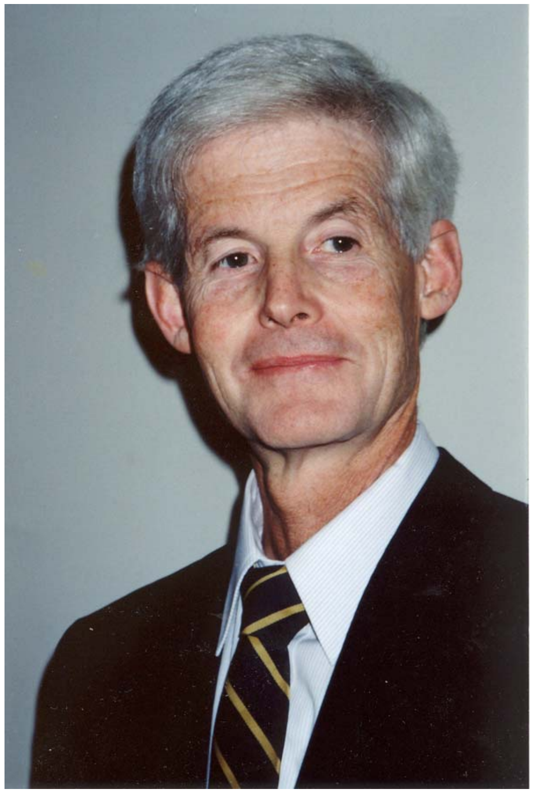


The Proof (concluded)

- Note that $c_1, c_2, \ldots, c_{|\, x\, |^k - 1}$ constitute the variables of $R(x)$.
- The overall circuit $R(x)$ (on p. 323) is satisfiable if and only if there is a truth assignment B such that the computation table accepts.
- This happens if and only if M accepts x, i.e., $x \in L$.

Stephen Arthur Cook[a] (1939–)

Richard Karp, "It is to our everlasting shame that we were unable to persuade the math department [of UC-Berkeley] to give him tenure."



[a]Turing Award (1982). See http://conservancy.umn.edu/handle/107226 for an interview in 2002.

NP-Complete Problems

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

I predict that scientists will one day adopt a new principle: "NP-complete problems are hard." That is, solving those problems efficiently is impossible on any device that could be built in the real world, whatever the final laws of physics turn out to be.
— Scott Aaronson (2008)

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 38 (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*

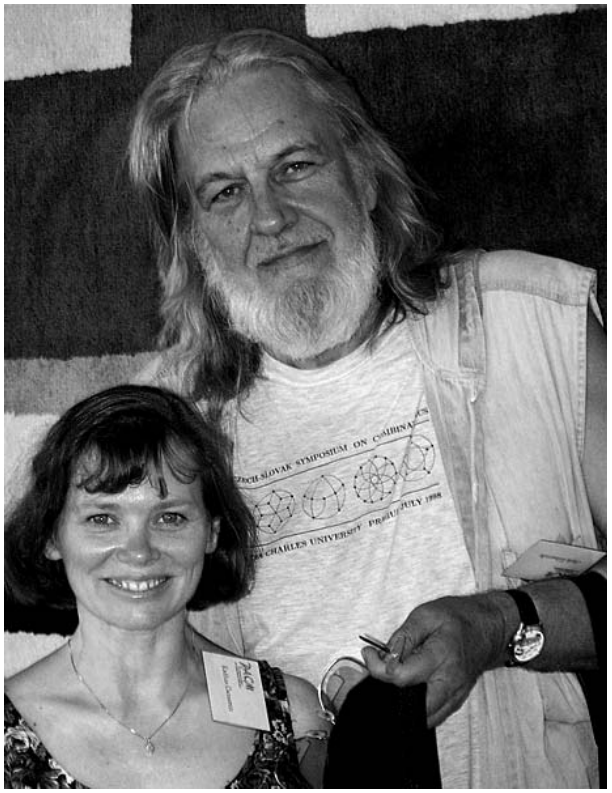
$$L = \{x : \exists y \, (x, y) \in R\}.$$

- Suppose such an R exists.
- L can be decided by this NTM:
  - On input x, the NTM guesses a y of length $\leq |\, x \,|^k$.
  - It then 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.
- R is polynomially balanced as N is polynomially bounded.
- R is polynomially decidable because it can be efficiently verified by consulting N's transition function.
- Finally $L = \{x : (x, y) \in R \text{ for some } y\}$ because N decides L.

Jack Edmonds (1934–)


CZECH-SLOVAK SYMPOSIUM ON COMBINATORICS
CHARLES UNIVERSITY PRAGUE JULY 1998

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,[a] but verification must be easy.
- NP is the class of *easy-to-verify*[b] problems.

[a] Unless P equals NP.
[b] That is, in polynomial time.

Comments (concluded)

- The degree k is not an input.
- How to find the k needed by the NTM is of no concern.[a]
- We only need to prove there *exists* an NTM that accepts L in nondeterministic polynomial time.

[a]Contributed by Mr. Kai-Yuan Hou (`B99201038`, `R03922014`) on November 3, 2015.

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

- From Propositions 30 (p. 295) and Proposition 33 (p. 298), 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 that it works *for your thesis*).

I thought NP-completeness was an interesting idea:
I didn't quite realize its potential impact.
— Stephen Cook (1998)

I was indeed surprised by Karp's work
since I did not expect so many
wonderful problems were NP-complete.
— Leonid Levin (1998)

Correct Use of Reduction in Proving NP-Completeness

- Recall that L_1 reduces to L_2 if there is an efficient function R such that for all inputs x (p. 270),

$$x \in L_1 \text{ if and only if } R(x) \in L_2.$$

- When L_1 is known to be NP-complete and when $L_2 \in \text{NP}$, then L_2 is NP-complete.
- A common mistake is to focus on solving $x \in L_1$ or solving $R(x) \in L_2$.
- The correct way is to, given a certificate for $x \in L_1$ (a satisfying truth assignment, e.g.), construct a certificate for $R(x) \in L_2$ (a Hamiltonian path, e.g.), and vice versa.

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. 318) and the reduction of CIRCUIT SAT to SAT (p. 284).
- The resulting CNF has at most 3 literals for each clause.
  - This accidentally 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.
  - So

$$x_1 \lor x_2 \quad \text{becomes} \quad x_1 \lor x_2 \lor x_2.$$

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 39 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.)*

The Proof (continued)

- 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.
  - $x_1, x_2, \ldots, x_k$ 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.
  - It is logically equivalent to
$$x_1 \Rightarrow x_2 \Rightarrow \cdots \Rightarrow x_k \Rightarrow x_1.$$
  - So $x_1, x_2, \ldots, x_k$ must assume an identical truth value for the whole expression to be satisfied.
- Note that each clause $\neg x_i \lor x_j$ above has only 2 literals.
- The resulting equivalent expression satisfies the conditions for x.

An Example

- Suppose we are given the following 3SAT expression

$$\cdots(\neg x \lor w \lor g) \land \cdots \land (x \lor y \lor z) \cdots .$$

- The transformed expression is

$$\cdots(\boxed{\neg x_1} \lor w \lor g) \land \cdots \land (x_2 \lor y \lor z) \cdots (\boxed{\neg x_1} \lor x_2) \land (\neg x_2 \lor \boxed{x_1}).$$

  - Variable x_1 appears 3 times.
  - Literal x_1 appears once.
  - Literal $\neg x_1$ appears 2 times.

2SAT Is in NL $\subseteq$ P

- Let ϕ be an instance of 2SAT: Each clause has 2 literals.
- NL is a subset of P (p. 249).
- By Eq. (2) on p. 262, coNL equals NL.
- We need to show only that recognizing *unsatisfiable* 2SAT expressions is in NL.
- See the textbook for the complete proof.

Generalized 2SAT: MAX2SAT

- Consider a 2SAT formula.
- Let $K \in \mathbb{N}$.
- MAX2SAT asks whether there is a truth assignment that satisfies at least K of the clauses.
  - MAX2SAT becomes 2SAT when K equals the number of clauses.

Generalized 2SAT: MAX2SAT (concluded)

- MAX2SAT is an optimization problem.
  - With binary search, one can nail the maximum number of satisfiable clauses of 2SAT formulas.
- MAX2SAT $\in$ NP: Guess a truth assignment and verify the count.
- We now reduce 3SAT to MAX2SAT.

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.
- The clauses are symmetric with respect to x, y, and z.
- How many clauses can we satisfy?

[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)

- A truth assignment that satisfies $x \vee y \vee z$ can be *extended* to satisfy 7 of the 10 clauses of $r(x, y, z, w)$, *and no more*.
- A truth assignment that does *not* satisfy $x \vee y \vee z$ can be extended to satisfy only 6 of them, *and no more*.
- 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.
- Finally, set $K = 7m$.

The Proof (continued)

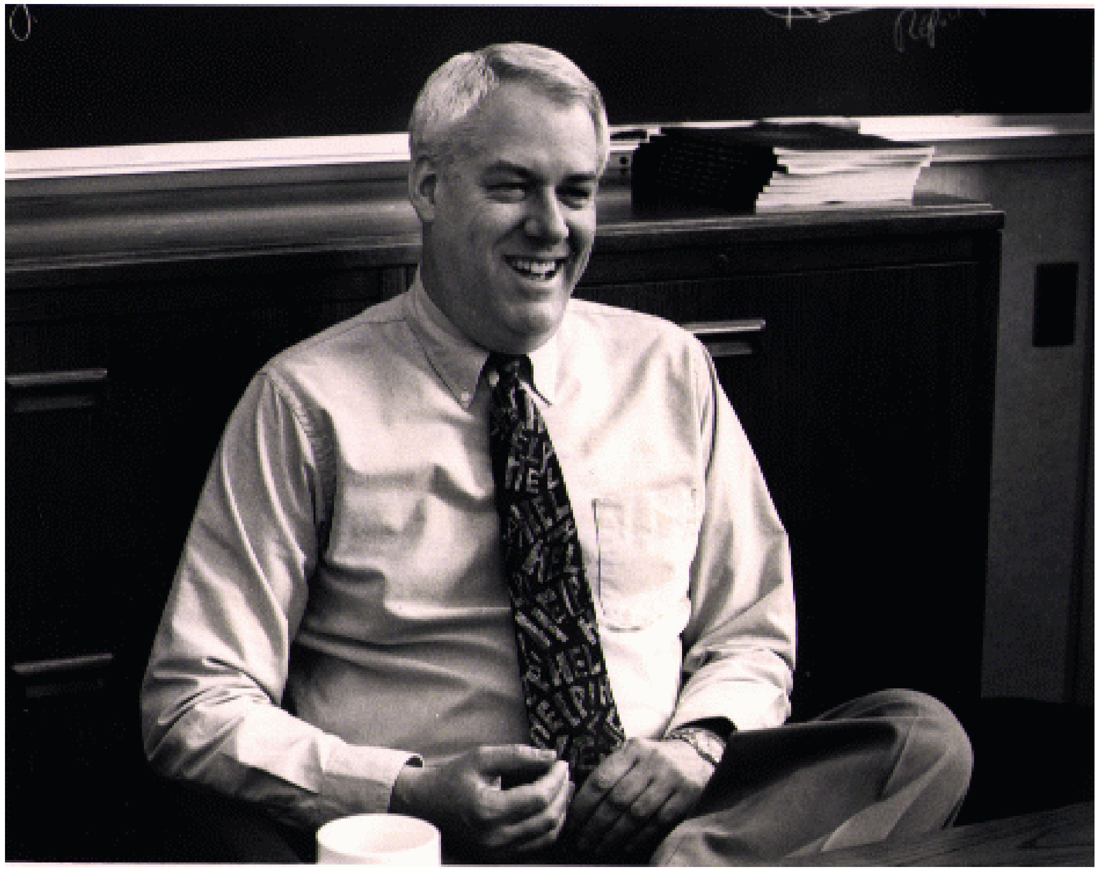
- We now show that K clauses of $R(\phi)$ can be satisfied if and only if ϕ is satisfiable.
- Suppose $K = 7m$ clauses of $R(\phi)$ can be satisfied.
  - 7 clauses of each group $r(\alpha, \beta, \gamma, w_i)$ must be satisfied because each group can have at most 7 clauses satisfied.[a]
  - Hence each clause $C_i = (\alpha \lor \beta \lor \gamma)$ of ϕ is satisfied by the same truth assignment.
  - So ϕ is satisfied.

[a]If 70% of the world population are male and if at most 70% of each country's population are male, then each country must have exactly 70% male population.

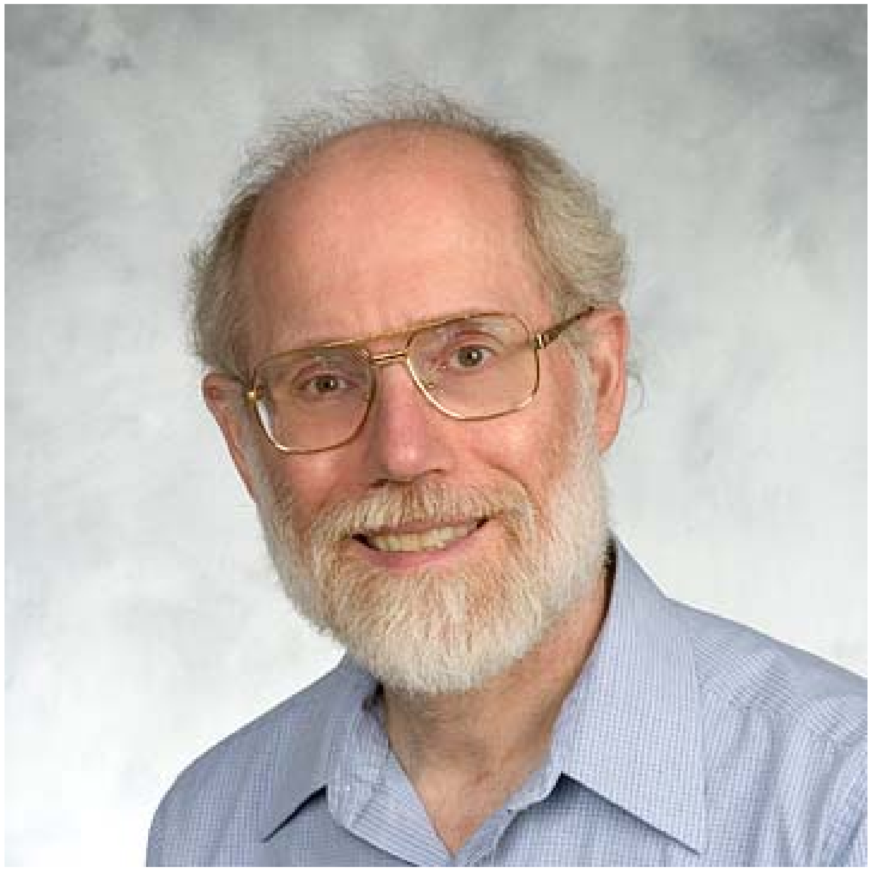
The Proof (concluded)

- Suppose ϕ is satisfiable.
  - Let T satisfy all clauses of ϕ.
  - Each group $r(\alpha, \beta, \gamma, w_i)$ can set its w_i appropriately to have 7 clauses satisfied.
  - So $K = 7m$ clauses are satisfied.

Michael R. Garey (1945–)



David S. Johnson (1945–)



Larry Stockmeyer (1948–2004)

NAESAT

- The NAESAT (for “not-all-equal” SAT) is like 3SAT.
- But there must be a satisfying truth assignment under which no clauses have all three literals equal in truth value.
- Equivalently, there is a truth assignment such that each clause has a literal assigned true and a literal assigned false.
- Equivalently, there is a satisfying truth assignment under which each clause has a literal assigned false.

NAESAT Is NP-Complete[a]

- Recall the reduction of CIRCUIT SAT to SAT on p. 284ff.
- It produced a CNF ϕ in which each clause has 1, 2, or 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 $\phi(z)$.
  - $\bar{T}$ takes the opposite truth value of T on every variable.
  - $\bar{T}$ also NAE-satisfies $\phi(z)$.
  - Under T or $\bar{T}$, variable z takes the value false.
  - This truth assignment $\mathcal{T}$ must satisfy all the clauses of ϕ.
    - $*$ Because z is not the reason that makes $\phi(z)$ true under $\mathcal{T}$ anyway.
  - So $\mathcal{T} \models \phi$.
  - And the original circuit is satisfiable.

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 $T(z) =$ `false` to obtain T'.
  - T' satisfies $\phi(z)$.
  - So in no clauses are all three literals false under T'.
  - In no clauses are all three literals true under T'.
    * Need to go over the detailed construction on p. 285 and p. 286.

Richard Karp[a] (1935–)

[a]Turing Award (1985).

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.
- Use $[\, i, j \,]$ to mean 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 there is no edge between any two nodes $i, j \in I$.
- INDEPENDENT SET: Given an undirected graph and a goal K, is there an independent set of size K?
- Many applications.

INDEPENDENT SET Is NP-Complete

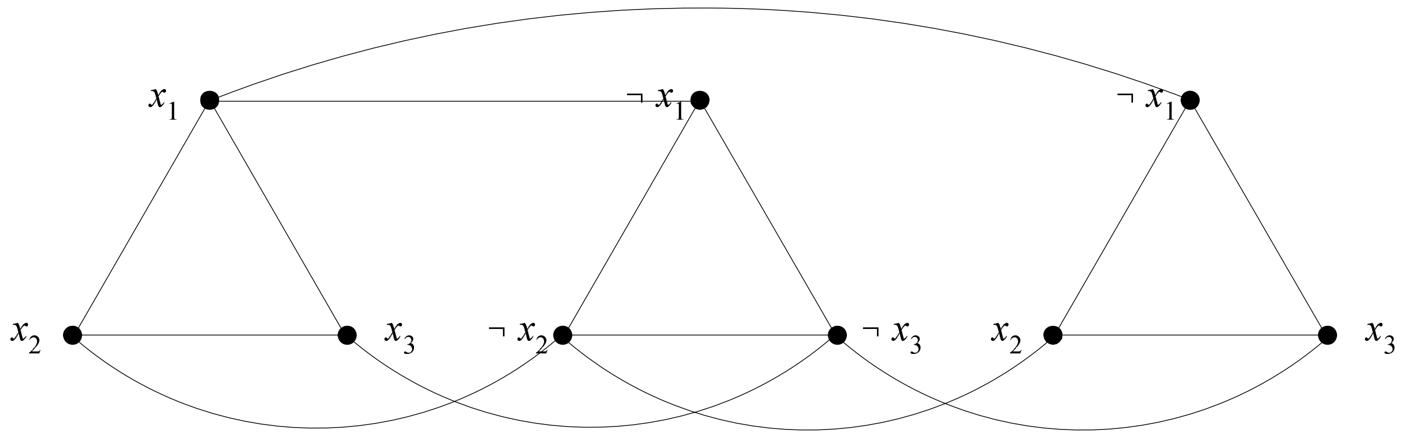
- This problem is in NP: Guess a set of nodes and verify that it is independent and meets the count.
- We will reduce 3SAT to INDEPENDENT SET.
- If a graph contains a triangle, any independent set can contain at most one node of the triangle.
- The results of the reduction will be graphs whose nodes can be partitioned into disjoint triangles, one for each clause.[a]

[a]Recall that the reduction does not have to be an onto function.

The Proof (continued)

- Let ϕ be a 3SAT formula with m clauses.
- We will construct graph G with $K = m$.
- Furthermore, ϕ is satisfiable if and only if G has an independent set of size K.
- Here is the reduction:
  - There is a triangle for each clause with the literals as the nodes.
  - Add edges between x and $\neg x$ for every variable x.

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



Same literal labels that appear in different clauses yield *distinct* nodes.

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.
  - So I 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 both cannot be in I.
  - T satisfies ϕ because it has a node from every triangle, thus satisfying every clause.[a]

[a]The variables without a truth value can be assigned arbitrarily. Contributed by Mr. Chun-Yuan Chen (`R99922119`) on November 2, 2010.

The Proof (concluded)

- Suppose ϕ is a satisfiable.
  - Let truth assignment T satisfy 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.
    - Because both literals x and $\neg x$ cannot be assigned true.

Other INDEPENDENT SET-Related NP-Complete Problems

Corollary 40 INDEPENDENT SET *is NP-complete for 4-degree graphs.*

Theorem 41 INDEPENDENT SET *is NP-complete for planar graphs.*

Theorem 42 (Garey and Johnson (1977)) INDEPENDENT SET *is NP-complete for 3-degree planar graphs.*

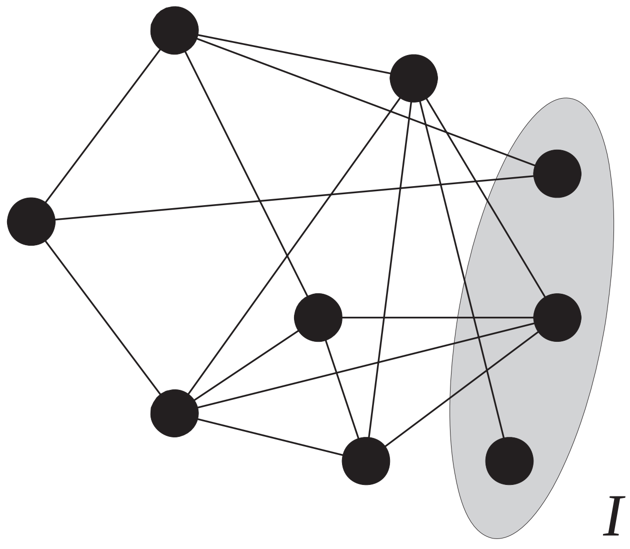
NODE COVER

- We are given an undirected graph G and a goal K.
- NODE COVER: Is there a set C with K or fewer nodes such that each edge of G has at least one of its endpoints (i.e., incident nodes) in C?
- Many applications.

NODE COVER Is NP-Complete[a]

Corollary 43 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.



[a]Karp (1972).

Remarks[a]

- Are INDEPENDENT SET and NODE COVER NP-complete if K is a constant?
  - No, because one can do an exhaustive search on all the possible node covers or independent sets (both $\binom{n}{K}$ of them, a polynomial).[b]
- Are INDEPENDENT SET and NODE COVER NP-complete if K is a linear function of n?
  - INDEPENDENT SET with $K = n/3$ and NODE COVER with $K = 2n/3$ remain NP-complete by our reductions.

[a]Contributed by Mr. Ching-Hua Yu (`D00921025`) on October 30, 2012.
[b]$n = |V|$.

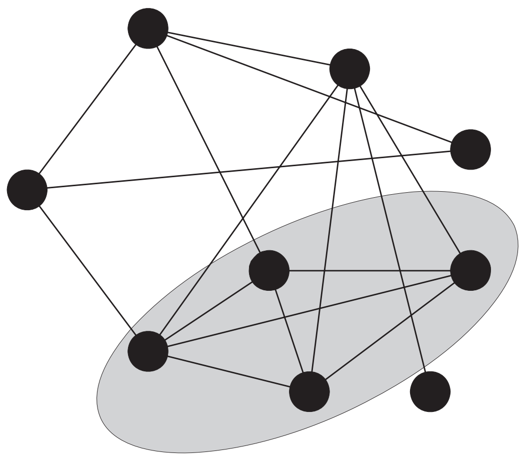
CLIQUE

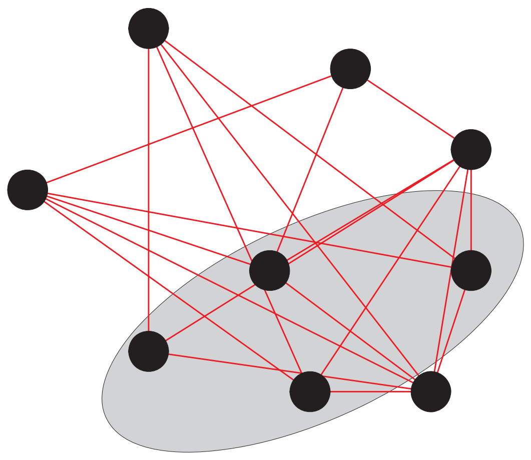
- We are given an undirected graph G and a goal K.
- CLIQUE asks if there is a set C with K nodes such that there is an edge between any two nodes $i, j \in C$.
- Many applications.

CLIQUE Is NP-Complete[a]

Corollary 44 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 a clique in $G \Leftrightarrow I$ is an independent set in $\bar{G}$.





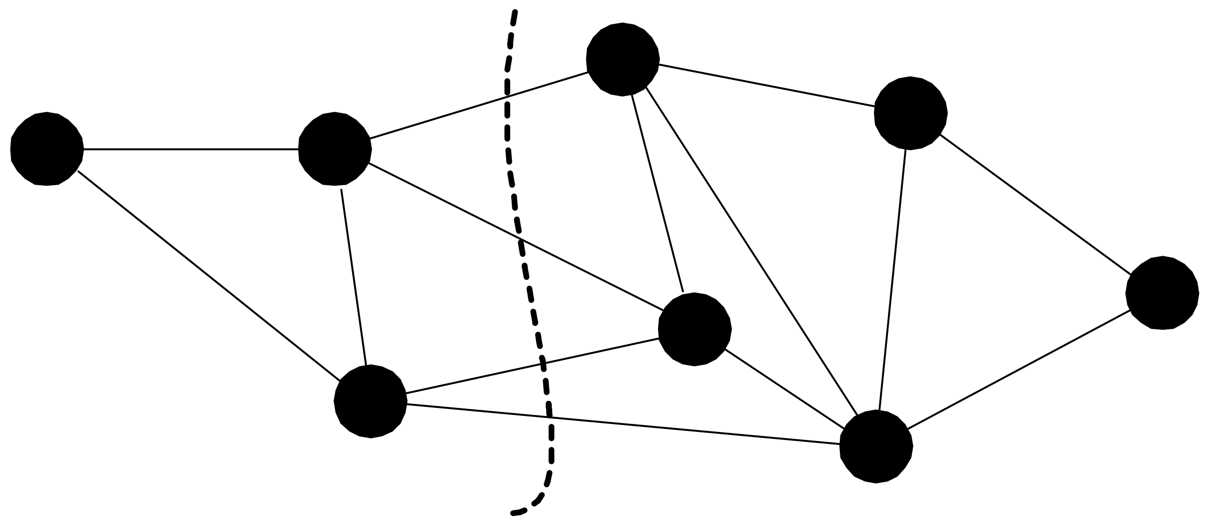
[a]Karp (1972).

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.[a]
- MAX CUT asks if there is a cut of size at least K.
  - K is part of the input.

[a]In time $O(|V| \cdot |E|)$ by Orlin (2012).

A Cut of Size 4



MIN CUT and MAX CUT (concluded)

- MAX CUT has applications in circuit 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); Mak and Wong (2000).