On P vs. NP

If 50 million people believe a foolish thing,
it's still a foolish thing.
— George Bernard Shaw (1856–1950)

Density[a]

The **density** of language $L \subseteq \Sigma^*$ is defined as

$$\text{dens}_L(n) = |\{x \in L : |\, x\,| \leq n\}|.$$

- If $L = \{0, 1\}^*$, then $\text{dens}_L(n) = 2^{n+1} - 1$.
- So the density function grows at most exponentially.
- For a unary language $L \subseteq \{0\}^*$,
  $$\text{dens}_L(n) \leq n + 1.$$
  - Because $L \subseteq \{\epsilon, 0, 00, \ldots, \overbrace{00 \cdots 0}^{n}, \ldots\}$.

[a] Berman and Hartmanis (1977).

Sparsity

- **Sparse languages** are languages with polynomially bounded density functions.
- **Dense languages** are languages with superpolynomial density functions.

Self-Reducibility for SAT

- An algorithm exhibits **self-reducibility** if it finds a certificate by exploiting algorithms for the *decision* version of the same problem.
- Let ϕ be a boolean expression in n variables $x_1, x_2, \ldots, x_n$.
- $t \in \{0,1\}^j$ is a **partial** truth assignment for $x_1, x_2, \ldots, x_j$.
- $\phi[\,t\,]$ denotes the expression after substituting the truth values of t for $x_1, x_2, \ldots, x_{|\,t\,|}$ in ϕ.

An Algorithm for SAT with Self-Reduction

We call the algorithm below with empty t.

if $|\,t\,| = n$ **then**
    return $\phi[\,t\,]$;
else
    return $\phi[\,t0\,] \lor \phi[\,t1\,]$;
end if

The above algorithm runs in exponential time, by visiting all the partial assignments (or nodes on a depth-n binary tree).[a]

[a] The same idea was used in the proof of Proposition 72 on p. 606.

NP-Completeness and Density[a]

Theorem 80 *If a unary language* $U \subseteq \{0\}^*$ *is NP-complete, then* $P = NP$*.*

- Suppose there is a reduction R from SAT to U.
- We use R to find a truth assignment that satisfies boolean expression ϕ with n variables if it is satisfiable.
- Specifically, we use R to prune the exponential-time exhaustive search on p. 750.
- The trick is to keep the already discovered results $\phi[\,t\,]$ in a table H.

[a]Berman (1978).

```
if |t| = n then
    return φ[t];
else
    if (R(φ[t]), v) is in table H then
        return v;
    else
        if φ[t0] = "satisfiable" or φ[t1] = "satisfiable" then
            Insert (R(φ[t]), "satisfiable") into H;
            return "satisfiable";
        else
            Insert (R(φ[t]), "unsatisfiable") into H;
            return "unsatisfiable";
        end if
    end if
end if
```

The Proof (continued)

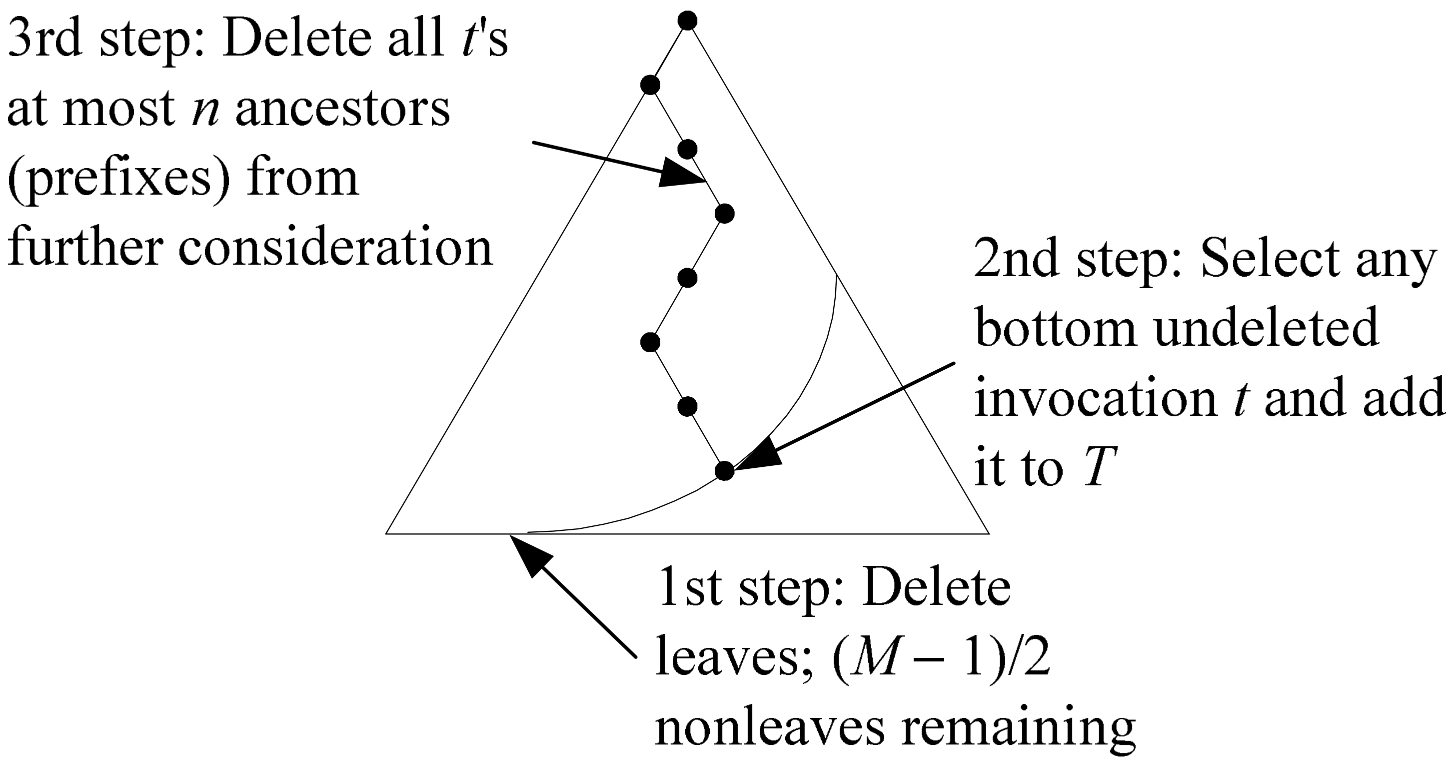
- Since R is a reduction, $R(\phi[\,t\,]) = R(\phi[\,t'\,])$ implies that $\phi[\,t\,]$ and $\phi[\,t'\,]$ must be both satisfiable or unsatisfiable.
- $R(\phi[\,t\,])$ has polynomial length $\leq p(n)$ because R runs in log space.
- As R maps to unary numbers, there are only polynomially many $p(n)$ values of $R(\phi[\,t\,])$.
- How many nodes of the complete binary tree (of invocations/truth assignments) need to be visited?

The Proof (continued)

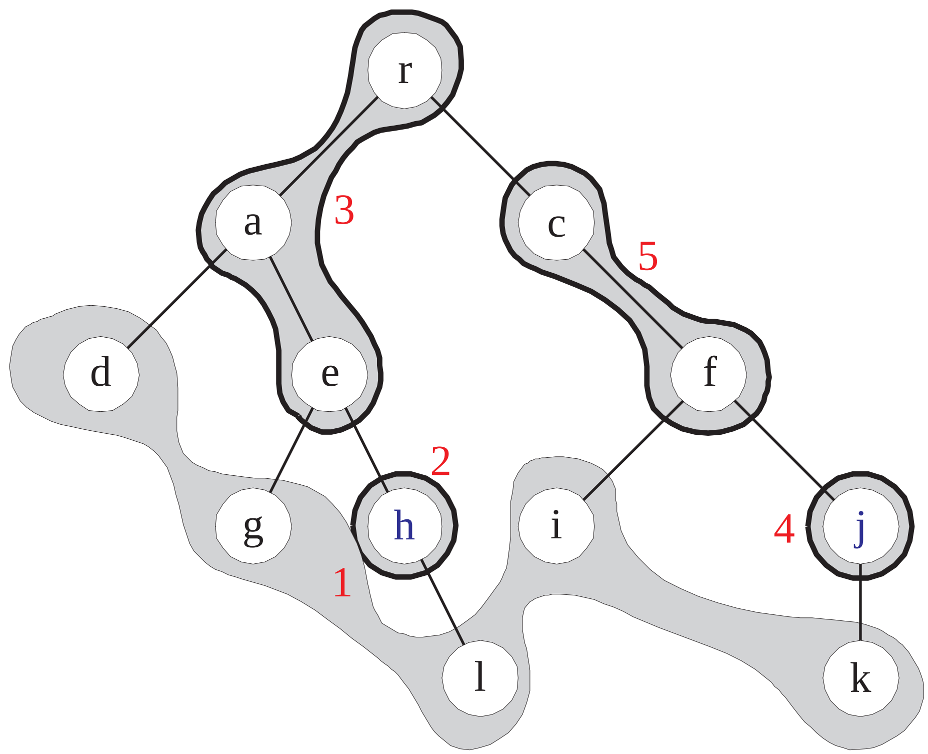
- A search of the table takes time $O(p(n))$ in the random-access memory model.
- The running time is $O(Mp(n))$, where M is the total number of invocations of the algorithm.
- If that number is a polynomial, the overall algorithm runs in polynomial time and we are done.
- The invocations of the algorithm form a binary tree of depth at most n.

The Proof (continued)

- There is a set $T = \{t_1, t_2, \ldots\}$ of invocations (partial truth assignments, i.e.) such that:
  1. $|T| \geq (M-1)/(2n)$.
  2. All invocations in T are recursive (nonleaves).
  3. None of the elements of T is a prefix of another.


3rd step: Delete all t's
at most n ancestors
(prefixes) from
further consideration
2nd step: Select any
bottom undeleted
invocation t and add
it to T
1st step: Delete
leaves; $(M-1)/2$
nonleaves remaining

An Example



$T = \{\, \text{h}, \text{j} \,\}$; none of h and j is a prefix of the other.

The Proof (continued)

- All invocations $t \in T$ have different $R(\phi[\,t\,])$ values.
  - The invocation of one started after the invocation of the other had terminated.
  - If they had the same value, the one that was invoked later would have looked it up, and therefore would not be recursive, a contradiction.
- The existence of T implies that there are at least $(M-1)/(2n)$ different $R(\phi[\,t\,])$ values in the table.

The Proof (concluded)

- We already know that there are at most $p(n)$ such values.
- Hence $(M-1)/(2n) \leq p(n)$.
- Thus $M \leq 2np(n)+1$.
- The running time is therefore $O(Mp(n)) = O(np^2(n))$.
- We comment that this theorem holds for any sparse language, not just unary ones.[a]

---

[a] Mahaney (1980).

coNP-Completeness and Density

Theorem 81 (Fortung (1979)) *If a unary language $U \subseteq \{0\}^*$ is coNP-complete, then P = NP.*

- Suppose there is a reduction R from SAT COMPLEMENT to U.
- The rest of the proof is basically identical except that, now, we want to make sure a formula is unsatisfiable.

The Power of Monotone Circuits

- Monotone circuits can only compute monotone boolean functions.
- They are powerful enough to solve a P-complete problem, MONOTONE CIRCUIT VALUE (p. 314).
- There are NP-complete problems that are not monotone; they cannot be computed by monotone circuits at all.
- There are NP-complete problems that are monotone; they can be computed by monotone circuits.
  - HAMILTONIAN PATH and CLIQUE.

$\text{CLIQUE}_{n,k}$

- $\text{CLIQUE}_{n,k}$ is the boolean function deciding whether a graph $G = (V, E)$ with n nodes has a clique of size k.
- The input gates are the $\binom{n}{2}$ entries of the adjacency matrix of G.
  - Gate g_{ij} is set to true if the associated undirected edge $\{ i, j \}$ exists.
- $\text{CLIQUE}_{n,k}$ is a monotone function.
- Thus it can be computed by a monotone circuit.
- This does not rule out that nonmonotone circuits for $\text{CLIQUE}_{n,k}$ may use fewer gates, however.

Crude Circuits

- One possible circuit for $\textsc{clique}_{n,k}$ does the following.
  1. For each $S \subseteq V$ with $|S| = k$, there is a circuit with $O(k^2)$ $\wedge$-gates testing whether S forms a clique.
  2. We then take an OR of the outcomes of all the $\binom{n}{k}$ subsets $S_1, S_2, \ldots, S_{\binom{n}{k}}$.
- This is a monotone circuit with $O(k^2 \binom{n}{k})$ gates, which is exponentially large unless k or $n - k$ is a constant.
- A **crude circuit** $\mathrm{CC}(X_1, X_2, \ldots, X_m)$ tests if *any* of $X_i \subseteq V$ forms a clique.
  - The above-mentioned circuit is $\mathrm{CC}(S_1, S_2, \ldots, S_{\binom{n}{k}})$.

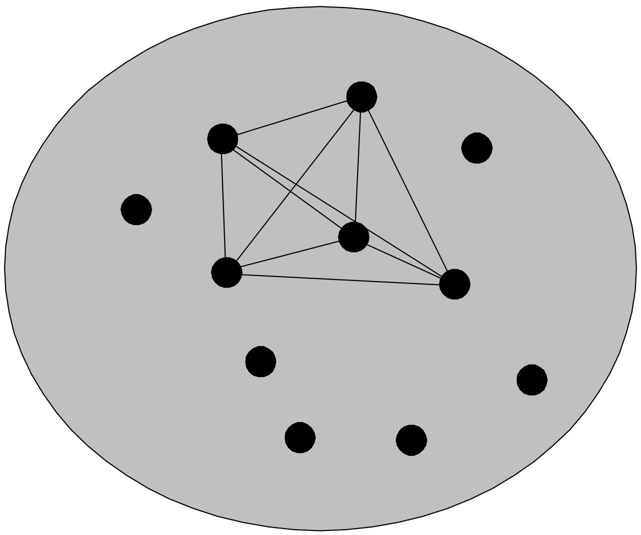
The Proof: Positive Examples

- Analysis will be applied to only **positive examples** and **negative examples** as inputs.
- A positive example is a graph that has $\binom{k}{2}$ edges connecting k nodes in all possible ways.
- There are $\binom{n}{k}$ such graphs.
- They all should elicit a true output from $\textsc{clique}_{n,k}$.

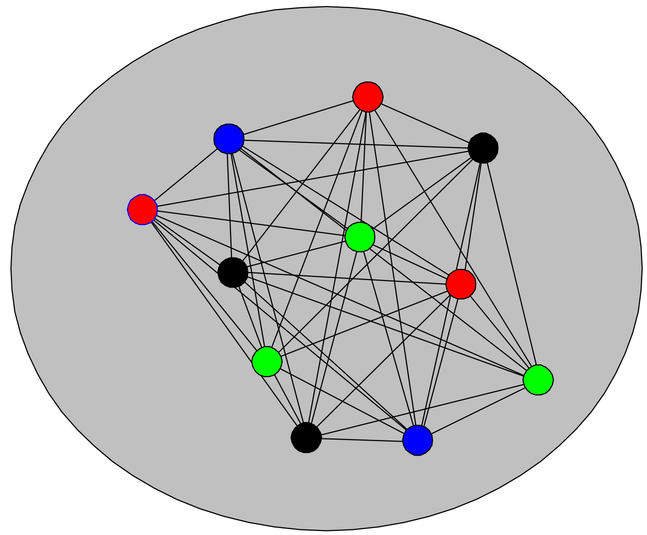
The Proof: Negative Examples

- Color the nodes with $k-1$ different colors and join by an edge any two nodes that are colored differently.
- There are $(k-1)^n$ such graphs.
- They all should elicit a false output from $\textsc{clique}_{n,k}$.
  - Each set of k nodes must have 2 identically colored nodes; hence there is no edge between them.

Positive and Negative Examples with $k = 5$



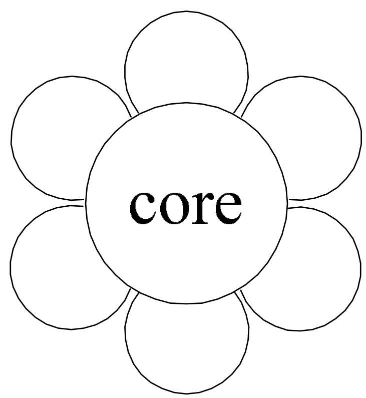
A positive example



A negative example

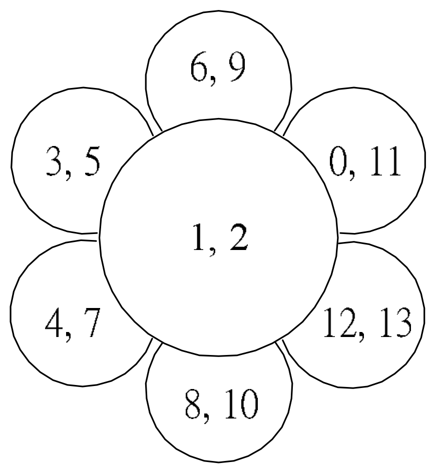
Sunflowers

- Fix $p \in \mathbb{Z}^+$ and $\ell \in \mathbb{Z}^+$.
- A **sunflower** is a family of p sets $\{P_1, P_2, \ldots, P_p\}$, called **petals**, each of cardinality at most ℓ.
- Furthermore, all pairs of sets in the family must have the same intersection (called the **core** of the sunflower).



A Sample Sunflower

$$\{\{1,2,3,5\},\{1,2,6,9\},\{0,1,2,11\},$$
$$\{1,2,12,13\},\{1,2,8,10\},\{1,2,4,7\}\}.$$



The Erdős-Rado Lemma

Lemma 82 *Let $\mathcal{Z}$ be a family of more than $M = (p-1)^{\ell}\ell!$ nonempty sets, each of cardinality ℓ or less. Then $\mathcal{Z}$ must contain a sunflower (with p petals).*

- Induction on ℓ.
- For $\ell = 1$, p different singletons form a sunflower (with an empty core).
- Suppose $\ell > 1$.
- Consider a *maximal* subset $\mathcal{D} \subseteq \mathcal{Z}$ of *disjoint* sets.
  - Every set in $\mathcal{Z} - \mathcal{D}$ intersects some set in $\mathcal{D}$.

The Proof of the Erdős-Rado Lemma (continued)

For example,

$$\begin{aligned}
\mathcal{Z} &= \{\{1,2,3,5\},\{1,3,6,9\},\{0,4,8,11\},\\
&\quad\ \{4,5,6,7\},\{5,8,9,10\},\{6,7,9,11\}\},\\
\mathcal{D} &= \{\{1,2,3,5\},\{0,4,8,11\}\}.
\end{aligned}$$

The Proof of the Erdős-Rado Lemma (continued)

- Suppose $\mathcal{D}$ contains at least p sets.
  - $\mathcal{D}$ constitutes a sunflower with an empty core.
- Suppose $\mathcal{D}$ contains fewer than p sets.
  - Let C be the union of all sets in $\mathcal{D}$.
  - $|\, C \,| < (p-1)\ell$.
  - C intersects every set in $\mathcal{Z}$ by $\mathcal{D}$'s maximality.
  - There is a $d \in C$ that intersects more than $\frac{M}{(p-1)\ell} = (p-1)^{\ell-1}(\ell-1)!$ sets in $\mathcal{Z}$.
  - Consider $\mathcal{Z}' = \{Z - \{d\} : Z \in \mathcal{Z}, d \in Z\}$.

The Proof of the Erdős-Rado Lemma (concluded)

- (continued)
  - $\mathcal{Z}'$ has more than $M' = (p-1)^{\ell-1}(\ell-1)!$ sets.
  - M' is just M with ℓ replaced with $\ell - 1$.
  - $\mathcal{Z}'$ contains a sunflower by induction, say
$$\{P_1, P_2, \ldots, P_p\}.$$
  - Now,
$$\{P_1 \cup \{d\}, P_2 \cup \{d\}, \ldots, P_p \cup \{d\}\}$$
is a sunflower in $\mathcal{Z}$.

Paul Erdős (1913–1996)

Comments on the Erdős-Rado Lemma

- A family of more than M sets must contain a sunflower.
- **Plucking** a sunflower means replacing the sets in the sunflower by its core.
- By *repeatedly* finding a sunflower and plucking it, we can reduce a family with more than M sets to a family with at most M sets.
- If $\mathcal{Z}$ is a family of sets, the above result is denoted by pluck($\mathcal{Z}$).
- Note: pluck($\mathcal{Z}$) is not unique.

An Example of Plucking

- Recall the sunflower on p. 768:

$$\begin{aligned}\mathcal{Z} &= \{\{1,2,3,5\},\{1,2,6,9\},\{0,1,2,11\},\\ &\quad \{1,2,12,13\},\{1,2,8,10\},\{1,2,4,7\}\}\end{aligned}$$

- Then

$$\text{pluck}(\mathcal{Z}) = \{\{1,2\}\}.$$

Razborov's Theorem

Theorem 83 (Razborov (1985)) *There is a constant c such that for large enough n, all monotone circuits for* $\textsc{clique}_{n,k}$ *with $k = n^{1/4}$ have size at least $n^{cn^{1/8}}$.*

- We shall approximate any monotone circuit for $\textsc{clique}_{n,k}$ by a restricted kind of crude circuit.
- The approximation will proceed in steps: one step for each gate of the monotone circuit.
- Each step introduces few errors (false positives and false negatives).
- But the final crude circuit has exponentially many errors.

The Proof

- Fix $k = n^{1/4}$.
- Fix $\ell = n^{1/8}$.
- Note that[a]
$$2\binom{\ell}{2} \leq k - 1.$$
- p will be fixed later to be $n^{1/8} \log n$.
- Fix $M = (p - 1)^{\ell} \ell!$.
  - Recall the Erdős-Rado lemma (p. 769).

[a]Corrected by Mr. Moustapha Bande (`D98922042`) on January 05, 2010.

The Proof (continued)

- Each crude circuit used in the approximation process is of the form $\mathrm{CC}(X_1, X_2, \ldots, X_m)$, where:
  - $X_i \subseteq V$.
  - $|X_i| \leq \ell$.
  - $m \leq M$.
- It answers true if any X_i is a clique.
- We shall show how to approximate any circuit for $\textsc{clique}_{n,k}$ by such a crude circuit, inductively.
- The induction basis is straightforward:
  - Input gate g_{ij} is the crude circuit $\mathrm{CC}(\{i, j\})$.

The Proof (continued)

- Any monotone circuit can be considered the OR or AND of two subcircuits.
- We shall show how to build approximators of the overall circuit from the approximators of the two subcircuits.
  - We are given two crude circuits $\mathrm{CC}(\mathcal{X})$ and $\mathrm{CC}(\mathcal{Y})$.
  - $\mathcal{X}$ and $\mathcal{Y}$ are two families of at most M sets of nodes, each set containing at most ℓ nodes.
  - We construct the approximate OR and the approximate AND of these subcircuits.
  - Then show both approximations introduce few errors.

The Proof: OR

- $\text{CC}(\mathcal{X} \cup \mathcal{Y})$ is *equivalent to* the OR of $\text{CC}(\mathcal{X})$ and $\text{CC}(\mathcal{Y})$.
  - A set of nodes $\mathcal{C} \in \mathcal{X} \cup \mathcal{Y}$ is a clique if and only if $\mathcal{C} \in \mathcal{X}$ is a clique or $\mathcal{C} \in \mathcal{Y}$ is a clique.
- Violations in using $\text{CC}(\mathcal{X} \cup \mathcal{Y})$ occur when $|\mathcal{X} \cup \mathcal{Y}| > M$.
- Such violations can be eliminated by using

$$\text{CC}(\text{pluck}(\mathcal{X} \cup \mathcal{Y}))$$

as the approximate OR of $\text{CC}(\mathcal{X})$ and $\text{CC}(\mathcal{Y})$.

The Proof: OR

- If $\mathrm{CC}(\mathcal{Z})$ is true, then $\mathrm{CC}(\mathrm{pluck}(\mathcal{Z}))$ must be true.
  - The quick reason: If Y is a clique, then a subset of Y must also be a clique.
  - For each $Y \in \mathcal{X} \cup \mathcal{Y}$, there must exist at least one $X \in \mathrm{pluck}(\mathcal{X} \cup \mathcal{Y})$ such that $X \subseteq Y$.
  - If Y is a clique, then this X is also a clique.
- We now bound the number of errors this approximate OR makes on the positive and negative examples.

The Proof: OR (concluded)

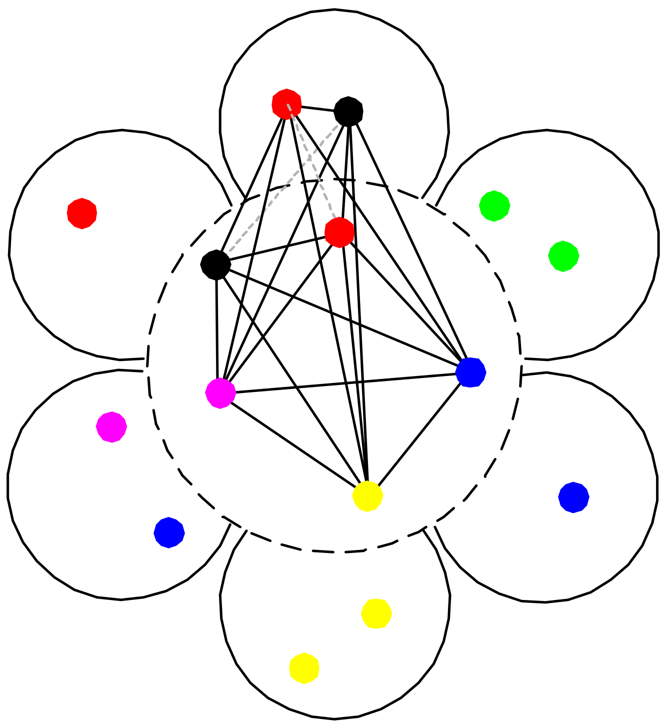
- $\text{CC}(\text{pluck}(\mathcal{X} \cup \mathcal{Y}))$ *introduces* a **false positive** if a negative example makes both $\text{CC}(\mathcal{X})$ and $\text{CC}(\mathcal{Y})$ return false but makes $\text{CC}(\text{pluck}(\mathcal{X} \cup \mathcal{Y}))$ return true.
- $\text{CC}(\text{pluck}(\mathcal{X} \cup \mathcal{Y}))$ *introduces* a **false negative** if a positive example makes either $\text{CC}(\mathcal{X})$ or $\text{CC}(\mathcal{Y})$ return true but makes $\text{CC}(\text{pluck}(\mathcal{X} \cup \mathcal{Y}))$ return false.
- How many false positives and false negatives are introduced by $\text{CC}(\text{pluck}(\mathcal{X} \cup \mathcal{Y}))$?

The Number of False Positives

Lemma 84 $\mathrm{CC}(\mathrm{pluck}(\mathcal{X} \cup \mathcal{Y}))$ *introduces at most* $\frac{M}{p-1}\, 2^{-p}(k-1)^n$ *false positives.*

- A plucking replaces the sunflower $\{Z_1, Z_2, \ldots, Z_p\}$ with its core Z.
- A false positive is *necessarily* a coloring such that:
  - There is a pair of identically colored nodes in each petal Z_i (and so both crude circuits return false).
  - But the core contains distinctly colored nodes.
    * This implies at least one node from each same-color pair was plucked away.
- We now count the number of such colorings.

Proof of Lemma 84 (continued)



Proof of Lemma 84 (continued)

- Color nodes V at random with $k-1$ colors and let $R(X)$ denote the event that there are repeated colors in set X.
- Now $\text{prob}[R(Z_1) \wedge \cdots \wedge R(Z_p) \wedge \neg R(Z)]$ is at most

$$\begin{aligned} & \text{prob}[R(Z_1) \wedge \cdots \wedge R(Z_p) | \neg R(Z)] \\ = \; & \prod_{i=1}^{p} \text{prob}[R(Z_i) | \neg R(Z)] \leq \prod_{i=1}^{p} \text{prob}[R(Z_i)]. \end{aligned} \quad (20)$$

  - First equality holds because $R(Z_i)$ are independent given $\neg R(Z)$ as Z contains their only common nodes.
  - Last inequality holds as the likelihood of repetitions in Z_i decreases given no repetitions in $Z \subseteq Z_i$.

Proof of Lemma 84 (continued)

- Consider two nodes in Z_i.
- The probability that they have identical color is $\frac{1}{k-1}$.
- Now $\text{prob}[\, R(Z_i)\,] \leq \frac{\binom{|Z_i|}{2}}{k-1} \leq \frac{\binom{\ell}{2}}{k-1} \leq \frac{1}{2}$.
- So the probability[a] that a random coloring is a new false positive is at most 2^{-p} by inequality (20).
- As there are $(k-1)^n$ different colorings, each plucking introduces at most $2^{-p}(k-1)^n$ false positives.

---

[a]Proportion, i.e.

Proof of Lemma 84 (concluded)

- Recall that $|\mathcal{X} \cup \mathcal{Y}| \leq 2M$.
- pluck($\mathcal{X} \cup \mathcal{Y}$) ends the moment the set system contains $\leq M$ sets.
- Each plucking reduces the number of sets by $p - 1$.
- Hence at most $\frac{M}{p-1}$ pluckings occur in pluck($\mathcal{X} \cup \mathcal{Y}$).
- At most
$$\frac{M}{p-1} \, 2^{-p}(k-1)^n$$
false positives are introduced.[a]

[a]Note that the numbers of errors are added not multiplied. Recall that we count how many *new* errors are introduced by each approximation step. Contributed by Mr. Ren-Shuo Liu (`D98922016`) on January 5, 2010.

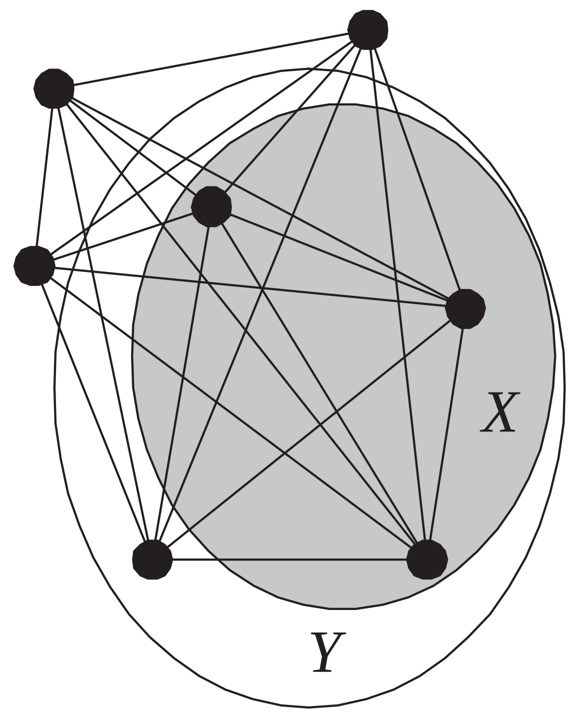
The Number of False Negatives

Lemma 85 CC(pluck($\mathcal{X} \cup \mathcal{Y}$)) *introduces no false negatives.*

- A plucking replaces sets in a crude circuit by their (common) subset.
- This makes the test for cliqueness less stringent (p. 781).[a]

[a] Recall that CC(pluck($\mathcal{X} \cup \mathcal{Y}$)) introduces a false negative if a positive example makes either CC($\mathcal{X}$) or CC($\mathcal{Y}$) return true but makes CC(pluck($\mathcal{X} \cup \mathcal{Y}$)) return false.

The Number of False Negatives (concluded)



The Proof: AND

- The approximate AND of crude circuits $\mathrm{CC}(\mathcal{X})$ and $\mathrm{CC}(\mathcal{Y})$ is

$$\mathrm{CC}(\mathrm{pluck}(\{X_i \cup Y_j : X_i \in \mathcal{X}, Y_j \in \mathcal{Y}, |X_i \cup Y_j| \leq \ell\})).$$

- We now count the number of errors this approximate AND makes on the positive and negative examples.

The Proof: AND (concluded)

- The approximate AND *introduces* a **false positive** if a negative example makes either $\mathrm{CC}(\mathcal{X})$ or $\mathrm{CC}(\mathcal{Y})$ return false but makes the approximate AND return true.
- The approximate AND *introduces* a **false negative** if a positive example makes both $\mathrm{CC}(\mathcal{X})$ and $\mathrm{CC}(\mathcal{Y})$ return true but makes the approximate AND return false.
- How many false positives and false negatives are introduced by the approximate AND?

The Number of False Positives

Lemma 86 *The approximate* AND *introduces at most* $M^2 2^{-p}(k-1)^n$ *false positives.*

- We prove this claim in stages.
- $\mathrm{CC}(\{X_i \cup Y_j : X_i \in \mathcal{X}, Y_j \in \mathcal{Y}\})$ introduces no false positives.
  - If $X_i \cup Y_j$ is a clique, both X_i and Y_j must be cliques, making both $\mathrm{CC}(\mathcal{X})$ and $\mathrm{CC}(\mathcal{Y})$ return true.
- $\mathrm{CC}(\{X_i \cup Y_j : X_i \in \mathcal{X}, Y_j \in \mathcal{Y}, |X_i \cup Y_j| \leq \ell\})$ introduces no additional false positives because we are testing fewer sets for cliqueness.

Proof of Lemma 86 (concluded)

- $|\{X_i \cup Y_j : X_i \in \mathcal{X}, Y_j \in \mathcal{Y}, |X_i \cup Y_j| \leq \ell\}| \leq M^2$.
- Each plucking reduces the number of sets by $p - 1$.
- So pluck$(\{X_i \cup Y_j : X_i \in \mathcal{X}, Y_j \in \mathcal{Y}, |X_i \cup Y_j| \leq \ell\})$ involves $\leq M^2/(p-1)$ pluckings.
- Each plucking introduces at most $2^{-p}(k-1)^n$ false positives by the proof of Lemma 84 (p. 783).
- The desired upper bound is

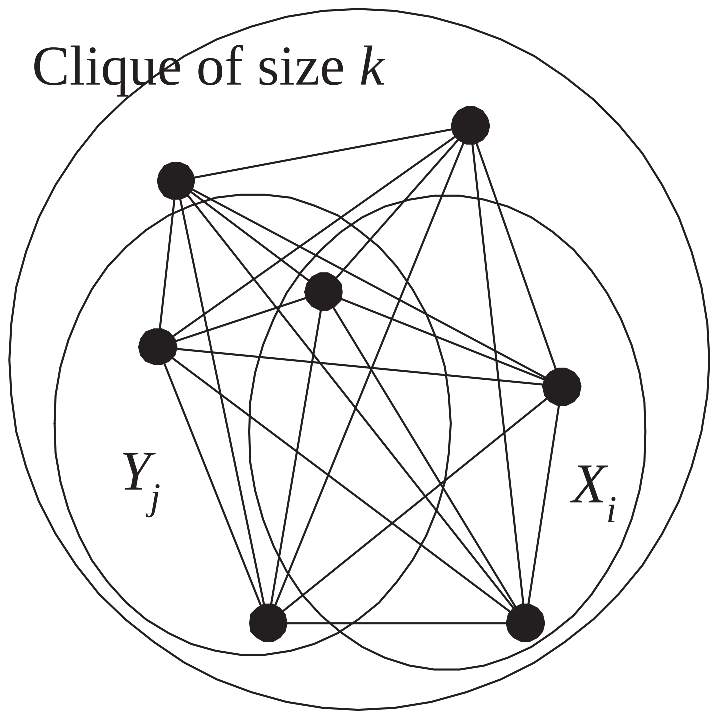
$$[\,M^2/(p-1)\,]\,2^{-p}(k-1)^n \leq M^2 2^{-p}(k-1)^n.$$

The Number of False Negatives

Lemma 87 *The approximate* AND *introduces at most* $M^2\binom{n-\ell-1}{k-\ell-1}$ *false negatives.*

- We again prove this claim in stages.
- $\mathrm{CC}(\{X_i \cup Y_j : X_i \in \mathcal{X}, Y_j \in \mathcal{Y}\})$ introduces no false negatives.
  - Suppose both $\mathrm{CC}(\mathcal{X})$ and $\mathrm{CC}(\mathcal{Y})$ accept a positive example with a clique of size k.
  - This clique must contain an $X_i \in \mathcal{X}$ and a $Y_j \in \mathcal{Y}$.
    * This is why both $\mathrm{CC}(\mathcal{X})$ and $\mathrm{CC}(\mathcal{Y})$ return true.
  - As this clique also contains $X_i \cup Y_j$, the new circuit returns true.

Proof of Lemma 87 (continued)



Proof of Lemma 87 (continued)

- $\mathrm{CC}(\{X_i \cup Y_j : X_i \in \mathcal{X}, Y_j \in \mathcal{Y}, |X_i \cup Y_j| \leq \ell\})$ introduces $\leq M^2 \binom{n-\ell-1}{k-\ell-1}$ false negatives.
  - Deletion of set $Z = X_i \cup Y_j$ larger than ℓ introduces false negatives *only if* Z is part of a clique.
  - There are $\binom{n-|Z|}{k-|Z|}$ such cliques.
    - It is the number of positive examples whose clique contains Z.
  - $\binom{n-|Z|}{k-|Z|} \leq \binom{n-\ell-1}{k-\ell-1}$ as $|Z| > \ell$.
  - There are at most M^2 such Zs.

Proof of Lemma 87 (concluded)

- Plucking introduces no false negatives.
  - Recall that if $\mathrm{CC}(\mathcal{Z})$ is true, then $\mathrm{CC}(\mathrm{pluck}(\mathcal{Z}))$ must be true (p. 781).

Two Summarizing Lemmas

From Lemmas 84 (p. 783) and 86 (p. 792), we have:

Lemma 88 *Each approximation step introduces at most* $M^2 2^{-p}(k-1)^n$ *false positives.*

From Lemmas 85 (p. 788) and 87 (p. 794), we have:

Lemma 89 *Each approximation step introduces at most* $M^2 \binom{n-\ell-1}{k-\ell-1}$ *false negatives.*

The Proof (continued)

- The above two lemmas show that each approximation step introduces "few" false positives and false negatives.
- We next show that the resulting crude circuit has "a lot" of false positives or false negatives.

The Final Crude Circuit

Lemma 90 *Every final crude circuit is:*

1. *Identically false—thus wrong on all positive examples.*
2. *Or outputs true on at least half of the negative examples.*

- Suppose it is not identically false.
- By construction, it accepts at least those graphs that have a clique on some set X of nodes, with $|X| \leq \ell$, which at $n^{1/8}$ is less than $k = n^{1/4}$.
- The proof of Lemma 84 (p. 783ff) shows that at least half of the colorings assign different colors to nodes in X.
- So half of the negative examples have a clique in X and are accepted.

The Proof (continued)

- Recall the constants on p. 777: $k = n^{1/4}$, $\ell = n^{1/8}$, $p = n^{1/8} \log n$, $M = (p-1)^\ell \ell! < n^{(1/3)n^{1/8}}$ for large n.
- Suppose the final crude circuit is identically false.
  - By Lemma 89 (p. 798), each approximation step introduces at most $M^2 \binom{n-\ell-1}{k-\ell-1}$ false negatives.
  - There are $\binom{n}{k}$ positive examples.
  - The original monotone circuit for $\text{CLIQUE}_{n,k}$ has at least

$$\frac{\binom{n}{k}}{M^2 \binom{n-\ell-1}{k-\ell-1}} \geq \frac{1}{M^2} \left(\frac{n-\ell}{k} \right)^\ell \geq n^{(1/12)n^{1/8}}$$

gates for large n.

The Proof (concluded)

- Suppose the final crude circuit is not identically false.
  - Lemma 90 (p. 800) says that there are at least $(k-1)^n/2$ false positives.
  - By Lemma 88 (p. 798), each approximation step introduces at most $M^2 2^{-p}(k-1)^n$ false positives
  - The original monotone circuit for $\text{CLIQUE}_{n,k}$ has at least

$$\frac{(k-1)^n/2}{M^2 2^{-p}(k-1)^n} = \frac{2^{p-1}}{M^2} \geq n^{(1/3)n^{1/8}}$$

gates.

Alexander Razborov (1963–)

$\mathsf{P} \neq \mathsf{NP}$ Proved?

- Razborov's theorem says that there is a monotone language in NP that has no polynomial monotone circuits.
- If we can prove that all monotone languages in P have polynomial monotone circuits, then $\mathrm{P} \neq \mathrm{NP}$.
- But Razborov proved in 1985 that some monotone languages in P have no polynomial monotone circuits!

Finis