## Comments

- Zero knowledge is a property of the prover.
  - It is the robustness of the prover against attempts of the verifier to extract knowledge via interaction.
  - The verifier may deviate arbitrarily (but in polynomial time) from the predetermined program.
  - A verifier cannot use the transcript of the interaction to convince a third-party of the validity of the claim.
  - The proof is hence not transferable.

## Comments (continued)

- Whatever a verifier can "learn" from the specified prover $P$ via the communication channel could as well be computed from the verifier alone.
- The verifier does not learn anything except "$x \in L$."
- For all practical purposes "whatever" can be done after interacting with a zero-knowledge prover can be done by just believing that the claim is indeed valid.
- Zero-knowledge proofs yield no knowledge in the sense that they can be constructed by the verifier who believes the statement, and yet these proofs do convince him.

## Comments (concluded)

- The "paradox" is resolved by noting that it is not the transcript of the conversation that convinces the verifier, but the fact that this conversation was held "on line."
- There is no zero-knowledge requirement when $x \notin L$.
- *Computational* zero-knowledge proofs are based on complexity assumptions.
- It is known that if one-way functions exist, then zero-knowledge proofs exist for all problems in NP.

## Zero-Knowledge Proof of Quadratic Residuosity

1: **for** $m = 1, 2, \ldots, \log_2 n$ **do**

2: Peggy chooses a random $v \in Z_n^*$ and sends $y = v^2 \bmod n$ to Victor;

3: Victor chooses a random bit $i$ and sends it to Peggy;

4: Peggy sends $z = u^i v \bmod n$, where $u$ is a square root of $x$; $\{u^2 \equiv x \bmod n.\}$

5: Victor checks if $z^2 \equiv x^i y \bmod n$;

6: **end for**

7: Victor accepts $x$ if Line 5 is confirmed every time;

## Analysis

- Assume extracting the square root of a quadratic residue modulo a product of two primes is hard without knowing the factors.
- Suppose $x$ is a quadratic nonresidue.
  - Peggy can answer only one of the two possible challenges.
    - Reason: $y$ is a quadratic residue if and only if $xy$ is a quadratic nonresidue.
  - So Peggy will be caught in any given round with probability one half.

## Analysis (continued)

- Suppose $x$ is a quadratic residue.
  - Peggy can answer all challenges.
  - So Victor will accept $x$.
- How about the claim of zero knowledge?
- The transcript between Peggy and Victor when $x$ is a quadratic residue can be generated without Peggy!
  - So interaction with Peggy is useless.

## Analysis (continued)

- Here is how.
- Suppose $x$ is a quadratic residue.
- In each round of interaction with Peggy, the transcript is a triplet $(y, i, z)$.
- We present an efficient algorithm Bob that generates $(y, i, z)$ with the same probability *without* accessing Peggy.

# Analysis (concluded)

1: Bob chooses a random $z \in Z_n^*$;

2: Bob chooses a random bit $i$;

3: Bob calculates $y = z^2 x^{-i} \bmod n$;

4: Bob writes $(y, i, z)$ into the transcript;

## Comments

- Bob cheats because $(y, i, z)$ is *not* generated in the same order as in the original transcript.
  - Bob picks Victor's challenge first.
  - Bob then picks Peggy's answer.
  - Bob finally patches the transcript.
  - So it is not the transcript that convinces Victor, but that conversation with Peggy is held "on line."
- The same holds even if the transcript was generated by a cheating Victor's interaction with (honest) Peggy, but we skip the details.

## Zero-Knowledge Proof of 3 Colorability[a]

**for** $i = 1, 2, \ldots, |E|^2$ **do**

  Peggy chooses a random permutation $\pi$ of the 3-coloring $\phi$;

  Peggy samples an encryption scheme randomly and sends $\pi(\phi(1)), \pi(\phi(2)), \ldots, \pi(\phi(|V|))$ encrypted to Victor;

  Victor chooses at random an edge $e \in E$ and sends it to Peggy for the coloring of the endpoints of $e$;

  **if** $e = (u, v) \in E$ **then**

    Peggy reveals the coloring of $u$ and $v$ and "proves" that they correspond to their encryption;

  **else**

    Peggy stops;

  **end if**

---

[a] Goldreich, Micali, and Wigderson (1986).

10: **if** the "proof" provided in Line 6 is not valid **then**
11: Victor rejects and stops;
12: **end if**
13: **if** $\pi(\phi(u)) = \pi(\phi(v))$ or $\pi(\phi(u)), \pi(\phi(v)) \not\in \{1, 2, 3\}$ **then**
14: Victor rejects and stops;
15: **end if**
16: **end for**
17: Victor accepts;

## Analysis

- If the graph is 3-colorable and both Peggy and Victor follow the protocol, then Victor always accepts.
- If the graph is not 3-colorable and Victor follows the protocol, then however Peggy plays, Victor will accept with probability $\leq (1 - m^{-1})^{m^2} \leq e^{-m}$, where $m = |\, E \,|$.
- Thus the protocol is valid.
- This protocol yields no knowledge to Victor as all he gets is a bunch of random pairs.
- The proof that the protocol is zero-knowledge to *any* verifier is more intricate.

## IP and PSPACE

- We next prove that coNP $\subseteq$ IP.
- Shamir in 1990 proved that IP equals PSPACE using similar ideas.

## Interactive Proof for Boolean Unsatisfiability

- A 3SAT formula is a conjunction of disjunctions of at most three literals.
- We shall present an interactive proof for boolean unsatisfiability.
- For any unsatisfiable 3SAT formula $\phi(x_1, x_2, \ldots, x_n)$, there is an interactive proof for the fact that it is unsatisfiable.
- Therefore, coNP $\subseteq$ IP.

# Arithmetization of Boolean Formulas

The idea is to arithmetize the boolean formula.

- T $\to$ positive integer
- F $\to 0$
- $x_i \to x_i$
- $\bar{x_i} \to 1 - x_i$
- $\vee \to +$
- $\wedge \to \times$
- $\phi(x_1, x_2, \ldots, x_n) \to \Phi(x_1, x_2, \ldots, x_n)$

## The Arithmetic Version

- A boolean formula is transformed into a multivariate polynomial $\Phi$.
- It is easy to verify that $\phi$ is unsatisfiable if and only if

$$\sum_{x_1=0,1} \sum_{x_2=0,1} \cdots \sum_{x_n=0,1} \Phi(x_1, x_2, \ldots, x_n) = 0.$$

- But the above seems to require exponential time.
- We turn to more intricate methods.

# Choosing the Field

- Suppose $\phi$ has $m$ clauses of length three each.
- Then $\Phi(x_1, x_2, \ldots, x_n) \leq 3^m$ for any truth assignment $(x_1, x_2, \ldots, x_n)$.
- Because there are at most $2^n$ truth assignments,

$$\sum_{x_1=0,1} \sum_{x_2=0,1} \cdots \sum_{x_n=0,1} \Phi(x_1, x_2, \ldots, x_n) \leq 2^n 3^m.$$

## Choosing the Field (concluded)

- By choosing a prime $q > 2^n 3^m$ and working modulo this prime, proving unsatisfiability reduces to proving that

$$\sum_{x_1=0,1} \sum_{x_2=0,1} \cdots \sum_{x_n=0,1} \Phi(x_1, x_2, \ldots, x_n) \equiv 0 \bmod q.$$

- Working under a *finite* field allows us to uniformly select a random element in the field.

## Binding Peggy

- Peggy has to find a sequence of polynomials that satisfy a number of restrictions.
- The restrictions are imposed by Victor: After receiving a polynomial from Peggy, Victor sets a new restriction for the next polynomial in the sequence.
- These restrictions guarantee that if $\phi$ is unsatisfiable, such a sequence can always be found.
- However, if $\phi$ is not unsatisfiable, any Peggy has only a small probability of finding such a sequence.
  - The probability is taken over Victor's coin tosses.

# The Algorithm

- Peggy and Victor both arithmetize $\phi$ to obtain $\Phi$;
- Peggy picks a prime $q > 2^n 3^m$ and sends it to Victor;
- Victor rejects and stops if $q$ is not a prime;
- Victor sets $v_0$ to 0;
- **for** $i = 1, 2, \ldots, n$ **do**
  - Peggy calculates $P_i^*(z) = \sum_{x_{i+1}=0,1} \cdots \sum_{x_n=0,1} \Phi(r_1, \ldots, r_{i-1}, z, x_{i+1}, \ldots, x_n)$;
  - Peggy sends $P_i^*(z)$ to Victor;
  - Victor rejects and stops if $P_i^*(0) + P_i^*(1) \not\equiv v_{i-1} \bmod q$ or $P_i^*(z)$'s degree exceeds $m$; $\{P_i^*(z)$ has at most $m$ clauses.$\}$
  - Victor uniformly picks $r_i \in Z_q$ and sets $v_i = P_i^*(r_i) \bmod q$;
  - Victor sends $r_i$ to Peggy;
- **end for**
- Victor accepts iff $\Phi(r_1, r_2, \ldots, r_n) \equiv v_n \bmod q$;

## Comments

- The following invariant is maintained by the algorithm:

$$P_i^*(0) + P_i^*(1) \equiv P_{i-1}^*(r_{i-1}) \bmod q \tag{11}$$

  for $1 \leq i \leq n$.

- The computation of $v_1, v_2, \ldots, v_n$ must rely on Peggy's supplied polynomials as Victor does not have the power to carry out the exponential-time calculations.

- But $\Phi(r_1, r_2, \ldots, r_n)$ in Step 12 is computed without relying on Peggy's polynomials.

# Completeness

- Suppose $\phi$ is unsatisfiable.
- For $i \geq 1$,

$$\begin{aligned}
& P_i^*(0) + P_i^*(1) \\
= & \sum_{x_i=0,1} \cdots \sum_{x_n=0,1} \Phi(r_1, \ldots, r_{i-1}, x_i, \ldots, x_n) \\
= & P_{i-1}^*(r_{i-1}) \\
\equiv & v_{i-1} \bmod q.
\end{aligned}$$

## Completeness (concluded)

- In particular at $i = 1$, because $\phi$ is unsatisfiable, we have

$$\begin{aligned} P_1^*(0) + P_1^*(1) &= \sum_{x_1=0,1} \cdots \sum_{x_n=0,1} \Phi(x_1, \ldots, x_n) \\ &\equiv v_0 \\ &= 0 \bmod q. \end{aligned}$$

- Finally, $v_n = P_n^*(r_n) = \Phi(r_1, r_2, \ldots, r_n)$.
- Because all the tests by Victor will pass, Victor will accept $\phi$.

## Soundness

- Suppose $\phi$ is not unsatisfiable.
- An honest Peggy following the protocol will fail after sending $P_1^*(z)$.
- We will show that if Peggy is dishonest in one round (by sending a polynomial other than $P_i^*(z)$), then with high probability she must be dishonest in the next round, too.
- In the last round (Step 12), her dishonesty is exposed.

## Soundness (continued)

- Let $P_i(z)$ represent the polynomial sent by Peggy in place of $P_i^*(z)$.
- Victor calculates $v_i = P_i(r_i) \bmod p$.
- In order to deceive Victor in the next round, round $i+1$, Peggy must use $r_1, r_2, \ldots, r_i$ to find a $P_{i+1}(z)$ of degree at most $m$ such that

$$P_{i+1}(0) + P_{i+1}(1) = v_i \bmod q$$

(see Step 8 of the algorithm on p. 526).

- And so on to the end, except that Peggy has no control over Step 12.

# A Key Claim

**Theorem 82** *If* $P_i^*(0) + P_i^*(1) \not\equiv v_{i-1} \bmod q$*, then either Victor rejects in the* $i$*th round, or* $P_i^*(r_i) \not\equiv v_i \bmod q$ *with probability at least* $1 - (m/q)$*, where the probability is taken over Victor's choices of* $r_i$*.*

- Remember that Victor has no way of knowing $P_i^*(r_i)$.
- Victor calculates $v_i$'s with $P_i(z)$s, claimed by the not necessarily trust-worthy Peggy as $P_i^*(z)$s.
- What Victor can do is to check for consistencies.

## The Proof of Theorem 82 (continued)

- If Peggy sends a $P_i(z)$ which equals $P_i^*(z)$, then

$$P_i(0) + P_i(1) = P_i^*(0) + P_i^*(1) \not\equiv v_{i-1} \bmod q,$$

and Victor rejects immediately.

- Suppose Peggy sends a $P_i(z)$ different from $P_i^*(z)$.
- If $P_i(z)$ does not pass Victor's test

$$P_i(0) + P_i(1) \equiv v_{i-1} \bmod q, \tag{12}$$

then Victor rejects and we are done, too.

# The Proof of Theorem 82 (concluded)

- Finally, assume $P_i(z)$ passes the test (12).
- Because $P_i(z) - P_i^*(z) \not\equiv 0$ is a polynomial of degree at most $m$, it has at most $m$ roots $r_i \in Z_q$, i.e.,

$$P_i^*(r_i) \equiv v_i \bmod q.$$

- Hence

$$P_i^*(r_i) \equiv v_i \bmod q$$

with probability at most $m/q$.

## Soundness (continued)

- Suppose Victor does not reject in any of the first $n$ rounds.
- As $\phi$ is not unsatisfiable,

$$P_1^*(0) + P_1^*(1) \not\equiv v_0 \bmod q.$$

- By Theorem 82 (p. 532) and the fact that Victor does not reject, we have $P_1^*(r_1) \not\equiv v_1 \bmod q$ with probability at least $1 - (m/q)$.
- Now by Eq. (11) on p. 527,

$$P_1^*(r_1) = P_2^*(0) + P_2^*(1) \not\equiv v_1 \bmod q.$$

## Soundness (concluded)

- Iterating on this procedure, we eventually arrive at

$$P_n^*(r_n) \not\equiv v_n \bmod q$$

with probability at least $(1 - m/q)^n$.

- As $P_n^*(r_n) = \Phi(r_1, r_2, \ldots, r_n)$, Victor's last test at Step 12 fails and he rejects.

- Altogether, Victor rejects with probability at least

$$[\,1 - (m/q)\,]^n > 1 - (nm/q) > 2/3$$

because $q > 2^n 3^m$.

## An Example

- $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3)$.
- The above is satisfied by assigning true to $x_1$.
- The arithmetized formula is

$$\Phi(x_1, x_2, x_3) = (x_1 + x_2 + x_3) \times [\, x_1 + (1 - x_2) + (1 - x_3) \,].$$

- Indeed, $\sum_{x_1=0,1} \sum_{x_2=0,1} \sum_{x_3=0,1} \Phi(x_1, x_2, x_3) = 16 \neq 0$.
- We have $n = 3$ and $m = 2$.
- A prime $q$ that satisfies $q > 2^3 \times 3^2 = 72$ is 73.

## An Example (continued)

- The table below is an execution of the algorithm in $Z_{73}$ *when Peggy follows the protocol.*

| $i$ | $P_i^*(z)$ | $P_i^*(0) + P_i^*(1)$ | $= v_{i-1}?$ | $r_i$ | $v_i$ |
|---|---|---|---|---|---|
| 0 | | | | | 0 |
| 1 | $4z^2 + 8z + 2$ | 16 | no | | |

- Victor therefore rejects $\phi$ early on at $i = 1$.

## An Example (continued)

- Now suppose Peggy does not follow the protocol.
- In order to deceive Victor, she comes up with fake polynomials $P_i(z)$'s from beginning to end.
- The table below is an execution of the algorithm.

| $i$ | $P_i(z)$ | $P_i(0) + P_i(1)$ | $= v_{i-1}$? | $r_i$ | $v_i$ |
|---|---|---|---|---|---|
| 0 | | | | | 0 |
| 1 | $8z^2 + 11z + 27$ | 0 | yes | 10 | 61 |
| 2 | $10z^2 + 9z + 21$ | 61 | yes | 4 | 71 |
| 3 | $z^2 + 2z + 34$ | 71 | yes | $r_3$ | $P_3(r_3)$ |

## An Example (concluded)

- Victor has been satisfied up to round 3.
- Finally at Step 12, Victor checks if

$$\Phi(10, 4, r_3) \equiv P_3(r_3) \bmod 73.$$

- It can be verified that the only choices of $r_3 \in \{ 0, 1, \ldots, 72 \}$ that can mislead Victor are 10 and 12.
- The probability of that happening is only $2/73$.

# An Example

- $(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2)$.
- The above is unsatisfiable.
- The arithmetized formula is

  $$\Phi(x_1, x_2) = (x_1 + x_2) \times (x_1 + 1 - x_2) \times (1 - x_1 + x_2) \times (2 - x_1 - x_2).$$

- Because $\Phi(x_1, x_2) = 0$ for any *boolean* assignment $\{0, 1\}^2$ to $(x_1, x_2)$, certainly

  $$\sum_{x_1=0,1} \sum_{x_2=0,1} \Phi(x_1, x_2) = 0.$$

- With $n = 2$ and $m = 4$, a prime $q$ that satisfies $q > 2^2 \times 3^4 = 4 \times 81 = 324$ is 331.

## An Example (concluded)

- The table below is an execution of the algorithm in $Z_{331}$.

| $i$ | $P_i^*(z)$ | $P_i^*(0) + P_i^*(1)$ | $= v_{i-1}$? | $r_i$ | $v_i$ |
|---|---|---|---|---|---|
| 0 | | | | | 0 |
| 1 | $z(z+1)(1-z)(2-z)$ | 0 | yes | 10 | 283 |
| | $+(z+1)z(2-z)(1-z)$ | | | | |
| 2 | $(10+z) \times (11-z)$ | 283 | yes | 5 | 46 |
| | $\times(-9+z) \times (-8-z)$ | | | | |

- Victor calculates $\Phi(10, 5) \equiv 46 \bmod 331$.
- As it equals $v_2 = 46$, Victor accepts $\phi$ as unsatisfiable.

## Objections to the Soundness Proof?[a]

- Based on the steps required of a cheating Peggy on p. 531, why must we go through so many rounds (in fact, $n$ rounds)?
- Why not just go directly to round $n$:
  - Victor sends $r_1, r_2, \ldots, r_{n-1}$ to Peggy.
  - Peggy returns with a (claimed) $P_n^*(z)$.
  - Victor accepts if and only if $\Phi(r_1, r_2, \ldots, r_{n-1}, r_n) \equiv P_n^*(r_n) \bmod q$ for a random $r_n \in Z_q$.

---

[a] Contributed by Ms. Emily Hou (D89011) and Mr. Pai-Hsuen Chen (R90008) on January 2, 2002.

## Objections to the Soundness Proof? (continued)

- Let us analyze the compressed proposal when $\phi$ is satisfiable.
- To succeed in foiling Victor, Peggy must find a polynomial $P_n(z)$ of degree $m$ such that

$$\Phi(r_1, r_2, \ldots, r_{n-1}, z) \equiv P_n(z) \bmod q.$$

- But this she is able to do: Just give the verifier the polynomial $\Phi(r_1, r_2, \ldots, r_{n-1}, z)$!
- What has happened?

## Objections to the Soundness Proof? (concluded)

- You need the intermediate rounds to "tie" Peggy up with a chain of claims.
- In the original algorithm on p. 526, for example, $P_n(z)$ is bound by the equality $P_n(0) + P_n(1) \equiv v_{n-1} \bmod q$ in Step 8.
- That $v_{n-1}$ is in turn derived by an earlier polynomial $P_{n-1}(z)$, which is in turn bound by $P_{n-1}(0) + P_{n-1}(1) \equiv v_{n-2} \bmod q$, and so on.

# 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 $L \subseteq \{0, 00, \ldots, \overbrace{00 \cdots 0}^{n}, \ldots\}$.

---

[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 exploits **self-reducibility** if it reduces the problem to the same problem with a smaller size.
- Let $\phi$ 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 $\phi$.

## 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.

## NP-Completeness and Density[a]

**Theorem 83** *If a unary language $U \subseteq \{0\}^*$ is NP-complete, then P = NP.*

- Suppose there is a reduction $R$ from SAT to $U$.
- We shall use $R$ to guide us in finding the truth assignment that satisfies a given boolean expression $\phi$ with $n$ variables if it is satisfiable.
- Specifically, we use $R$ to prune the exponential-time exhaustive search on p. 549.
- The trick is to keep the already discovered results $\phi[t]$ in a hash 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]), 1) into H;
            return "satisfiable";
        else
            Insert (R(φ[t]), 0) into H;
            return "unsatisfiable";
        end if
    end if
end if
```

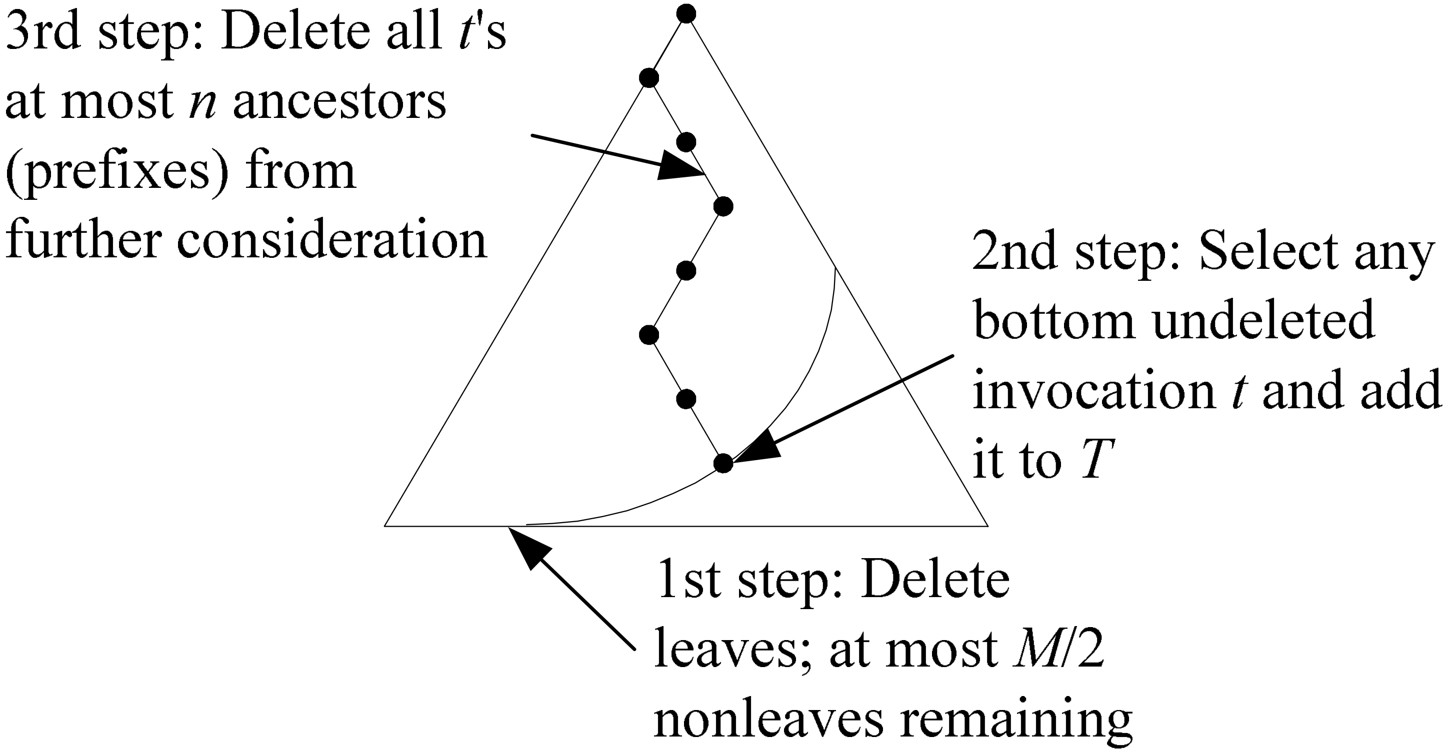
## The Proof (continued)

- 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?
- If that number is a polynomial, the overall algorithm runs in polynomial time and we are done.

## 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.
- The invocations of the algorithm form a binary tree of depth at most $n$.
- There is a set $T = \{t_1, t_2, \ldots\}$ of invocations (partial truth assignments, i.e.) such that:
  - $|T| \geq M/(2n)$.
  - All invocations in $T$ are **recursive** (nonleaves).
  - 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; at most M/2 nonleaves remaining

## The Proof (continued)

- All invocations $t \in T$ have different $R(\phi[\,t\,])$ values.
  - None of $s, t \in T$ is a prefix of another.
  - The invocation of one started after the invocation of the other had terminated.
  - If they had the same value, the one that was invoked second would have looked it up, and therefore would not be recursive, a contradiction.
- The existence of $T$ implies that there are at least $M/(2n)$ different $R(\phi[\,t\,])$ values in the table.

## The Proof (concluded)

- We already know that there are at most $p(n)$ such values.
- Hence $M/(2n) \leq p(n)$.
- Thus $M \leq 2np(n)$.
- 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).

## NP-Completeness and Density

**Theorem 84 (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.

*Finis*