Title: PSI 17/18 - Quantum Information - Lecture 5

Date: Feb 26, 2018 11:30 AM

URL: http://pirsa.org/18020048

Abstract:

Def: $\{0,1\}^*$ is the set of all bit strings of arbitrary length. A language is a subset of $\{0,1\}^*$.

[$x \in L$ if the associated function $f(x) = \text{TRUE}$, $x \notin L$ otherwise.]

x is a "yes instance" of L if $x \in L$, x is a "no instance" if $x \notin L$.

E.g.: PRIMES is the language $\{x \in \mathbb{Z}^+ \mid x \text{ is prime}\}$

Def: $\{0,1\}^*$ is the set of all bit strings of arbitrary length. A language is a subset of $\{0,1\}^*$.

[$x \in L$ if the associated function $f(x) =$ TRUE, $x \notin L$ otherwise.]

x is a "yes instance" of L if $x \in L$, x is a "no instance" if $x \notin L$.

E.g.: PRIMES is the language $\{x \in \mathbb{Z}^+ \mid x \text{ is prime}\}$

Worst-case complexity

Best algorithm

<u>Def:</u> A complexity class is a set of languages. The complexity class P is the set of languages that can be decided in time that is polynomial in the size of the instance.

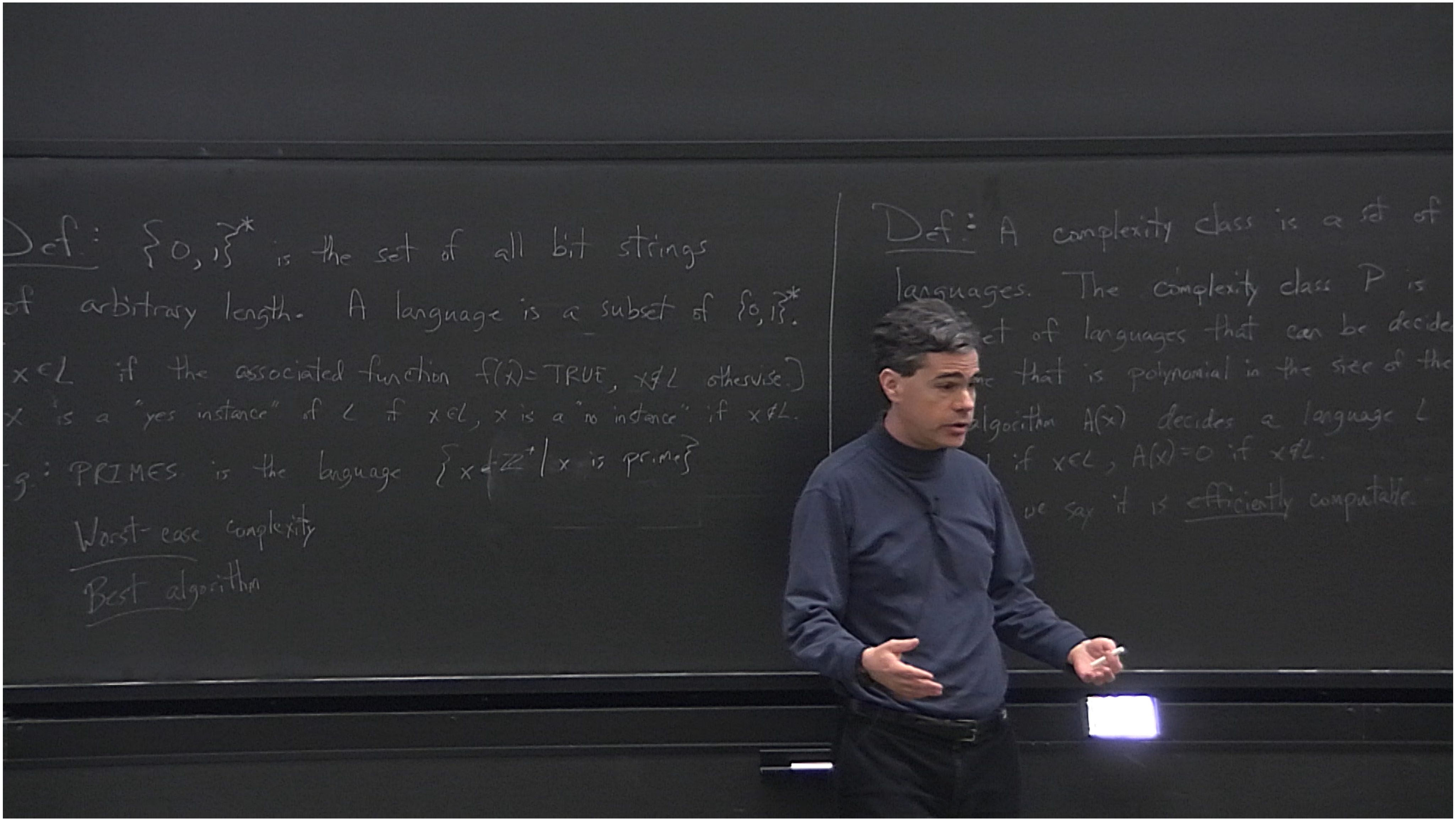
An algorithm $A(x)$ decides a language L if $A(x)=1$ if $x \in L$, $A(x)=0$ if $x \notin L$.

ed
e instance.
if

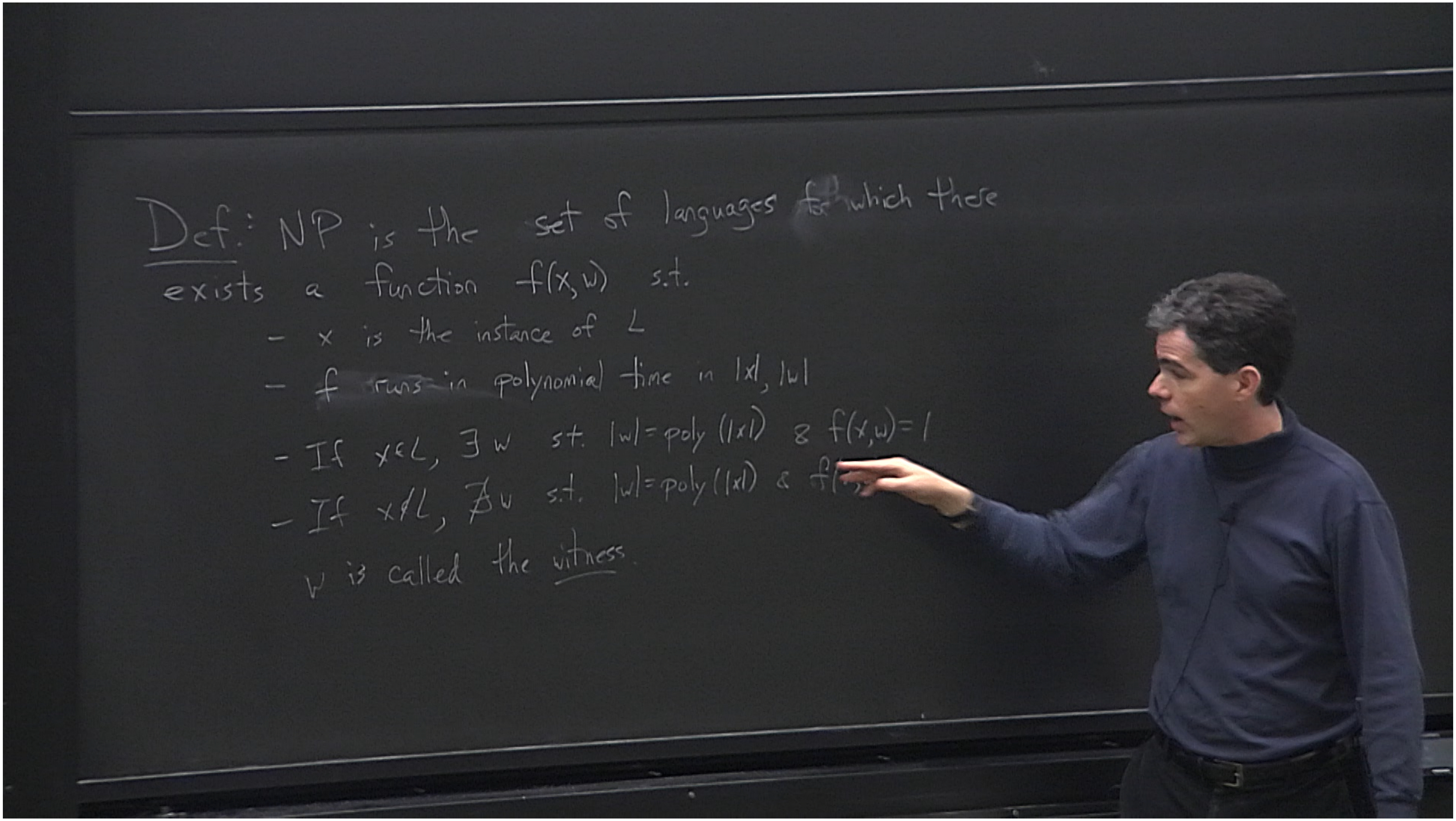
Halting problem $h(x) = 1$ if x halts, $h(x)$ if not.

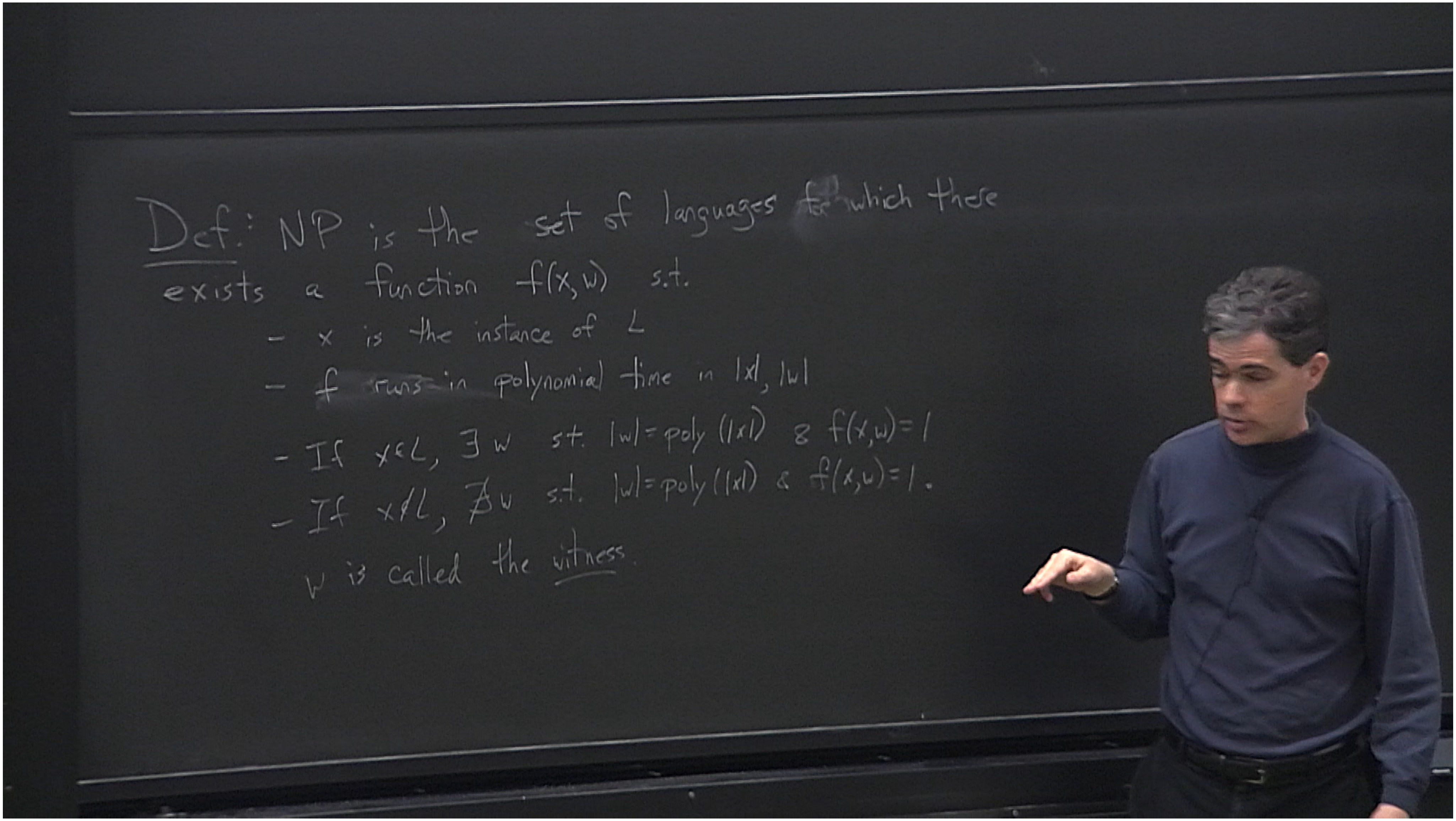
<u>Church-Turing thesis</u>: The set of computable problems is the same in every physically reasonable computational model.

<u>Strong Church-Turing thesis</u>: The complexity of languages differs by polynomial amounts between physically reasonable computational models.



Def: {0,1}* is the set of all bit strings
of arbitrary length. A language is a subset of {0,1}*.
x∈L if the associated function f(x)=TRUE, x∉L otherwise.]
x is a "yes instance" of L if x∈L, x is a "no instance" if x∉L.
PRIMES is the language {x∈Z+ | x is prime}
Worst-case complexity
Best algorithm
Def: A complexity class is a set of
languages. The complexity class P is
of languages that can be decide
that is polynomial in the size of the
algorithm A(x) decides a language L
if x∈L, A(x)=0 if x∉L.
we say it is efficiently computable.


Def: NP is the set of languages for which there exists a function f(x,w) s.t.
- x is the instance of L
- f runs in polynomial time in |x|, |w|
- If x∈L, ∃ w s.t. |w| = poly(|x|) & f(x,w) = 1
- If x∉L, ∄ w s.t. |w| = poly(|x|) &
w is called the witness.


Def.: NP is the set of languages for which there
exists a function $f(x,w)$ s.t.
- x is the instance of L
- f runs in polynomial time in $|x|, |w|$
- If $x \in L$, $\exists w$ s.t. $|w| = \mathrm{poly}(|x|)$ & $f(x,w)=1$
- If $x \notin L$, $\nexists w$ s.t. $|w| = \mathrm{poly}(|x|)$ & $f(x,w)=1$.
w is called the witness.

Example: 3-SAT (3-satisfiability)

Instances: Boolean formulas

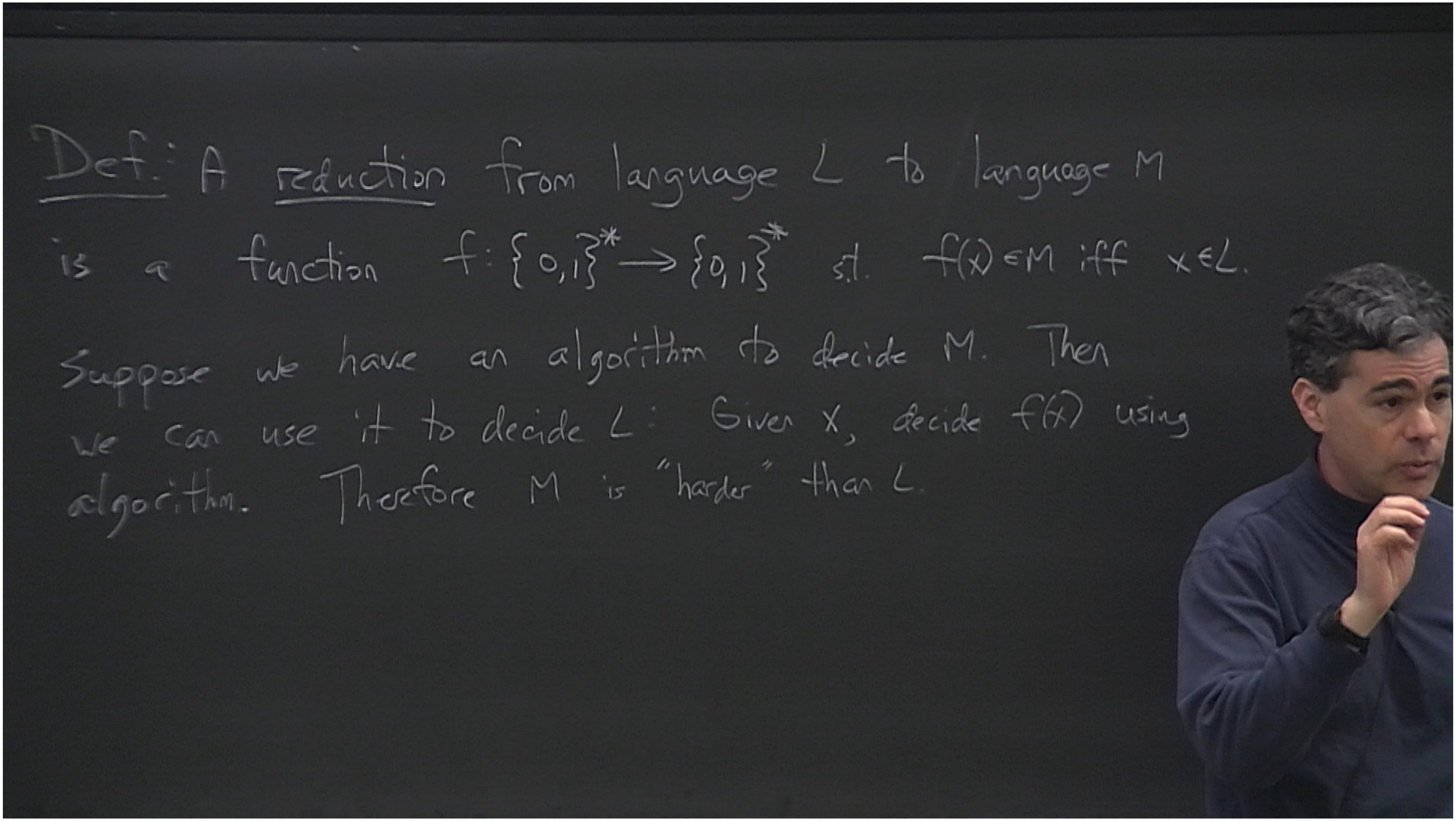
AND of clauses

Each clause is the OR of ≤ 3 variables or their negations

E.g. $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\overline{x_2} \vee \overline{x_4})$

formula $C \in$ 3-SAT if $\exists$ assignment of variables s.t. $C(x_0, x_1, \dots, x_{n-1}) = \text{TRUE}$

3-SAT $\in$ NP


Def.: A reduction from language L to language M
is a function f: {0,1}* → {0,1}* s.t. f(x) ∈ M iff x ∈ L.
Suppose we have an algorithm to decide M. Then
we can use it to decide L: Given x, decide f(x) using
algorithm. Therefore M is "harder" than L.

Def.: A <u>reduction</u> from language L to language M is a function $f: \{0,1\}^* \to \{0,1\}^*$ s.t. $f(x) \in M$ iff $x \in L$.

[Suppose we have an algorithm to decide M. Then we can use it to decide L: Given x, decide $f(x)$ using algorithm. Therefore M is "harder" than L.]

Use reductions s.t. f is efficiently computable.

A

there

i.e. $f(x,w)=0$ $\forall w$.

Example: 3-SAT (3-satisfiability)

Instances: Boolean formulas

AND of clauses

Each clause is the OR of ≤ 3 variables or their negations

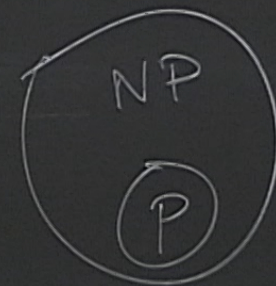
E.g. $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\overline{x_2} \vee \overline{x_4})$

formula $C \in$ 3-SAT if $\exists$ assignment of variables s.t. $C(x_0, x_1, \dots, x_{n-1}) =$ TRUE

3-SAT $\in$ NP

3-SAT is NP-complete (Cook-Levin thm.)

A language M is NP-complete if $M \in NP$ and $\forall L \in NP$ $\exists$ polytime reduction from L to M.



$P \stackrel{?}{=} NP$