15-453

FORMAL LANGUAGES, AUTOMATA AND COMPUTABILITY

NP-COMPLETENESS:
THE COOK-LEVIN THEOREM

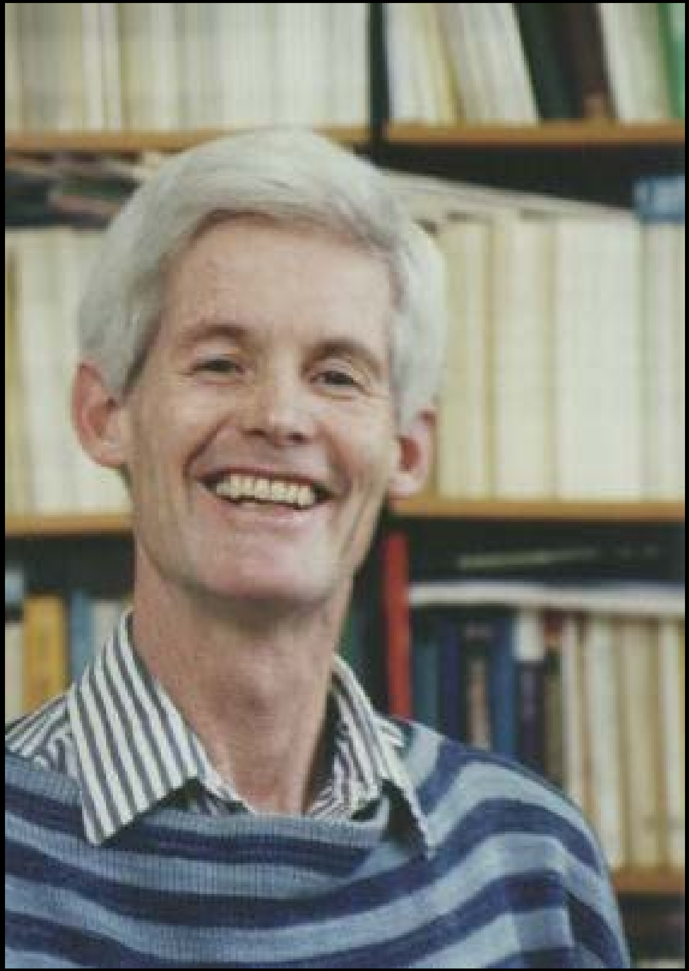
TUESDAY March 25

Theorem (Cook-Levin): SAT is NP-complete

Corollary: SAT $\in$ P if and only if P = NP

Theorem (Cook/Levin'71) P= NP $\Leftrightarrow$ SAT $\in$ P

Leonid Levin



Steve Cook

SAT = { ϕ | ϕ is a satisfiable boolean formula }

3-SAT = { ϕ | ϕ is a satisfiable 3cnf-formula }

A 3cnf-formula is of the form:

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (x_4 \vee x_2 \vee x_5) \wedge (x_3 \vee \neg x_2 \vee \neg x_1)$$

clauses

SAT = { ϕ | ϕ is a satisfiable boolean-formula }

3-SAT = { ϕ | ϕ is a satisfiable 3cnf-formula }

SAT, 3-SAT $\in$ NP (why?)

Theorem (Cook-Levin): SAT is NP-complete

Proof:

(1) SAT $\in$ NP (3SAT $\in$ NP)

(2) Every language A in NP is polynomial time reducible to SAT

Theorem (Cook-Levin): SAT is NP-complete

Proof:

(1) SAT $\in$ NP (3SAT $\in$ NP)

(2) Every language A in NP is polynomial time reducible to SAT

We build a poly-time reduction from A to SAT

The reduction turns a string w into a 3-cnf formula ϕ such that w $\in$ A iff ϕ $\in$ 3-SAT.

ϕ will *simulate* the NP machine N for A on w.

Theorem (Cook-Levin): SAT is NP-complete

Proof:

(1) SAT $\in$ NP (3SAT $\in$ NP)

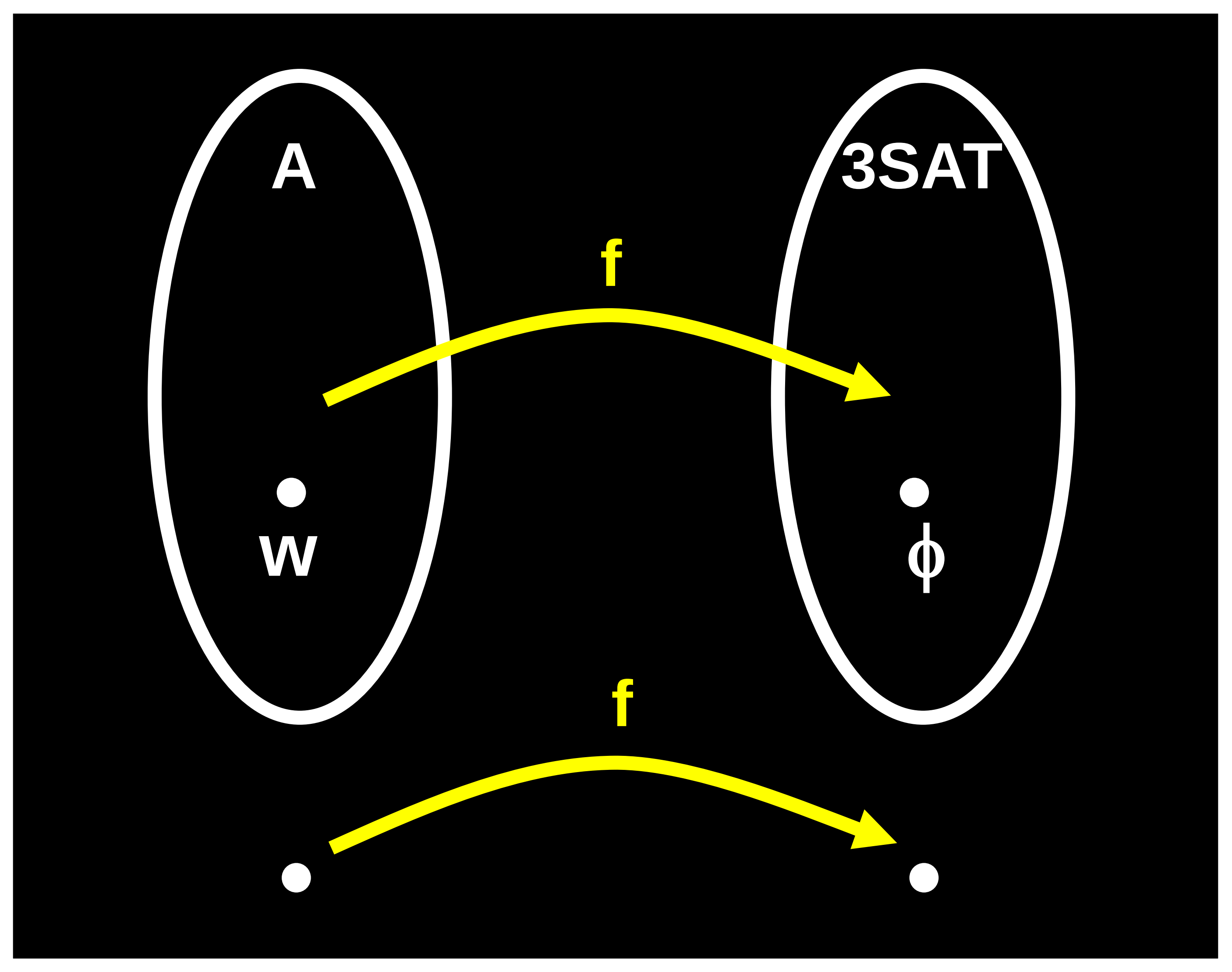
(2) Every language A in NP is polynomial time reducible to SAT

We build a poly-time reduction from A to SAT

The reduction turns a string w into a 3-cnf formula ϕ such that w $\in$ A iff $\phi \in$ 3-SAT.

ϕ will *simulate* the NP machine N for A on w.

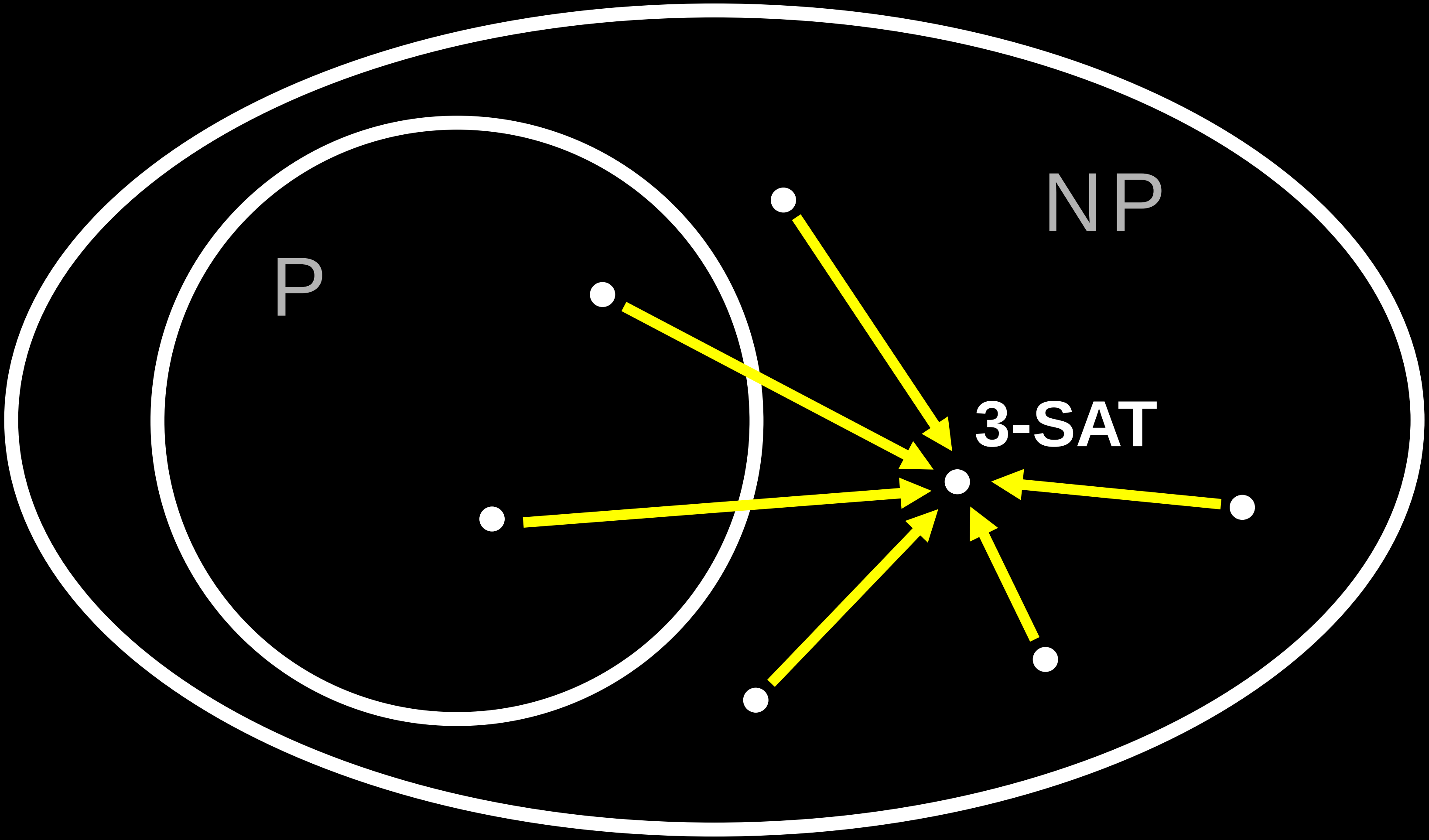
Let N be a non-deterministic TM that decides A in time n^k



The reduction f turns a string w into a 3-cnf formula ϕ such that: $w \in A \Leftrightarrow \phi \in 3SAT$.

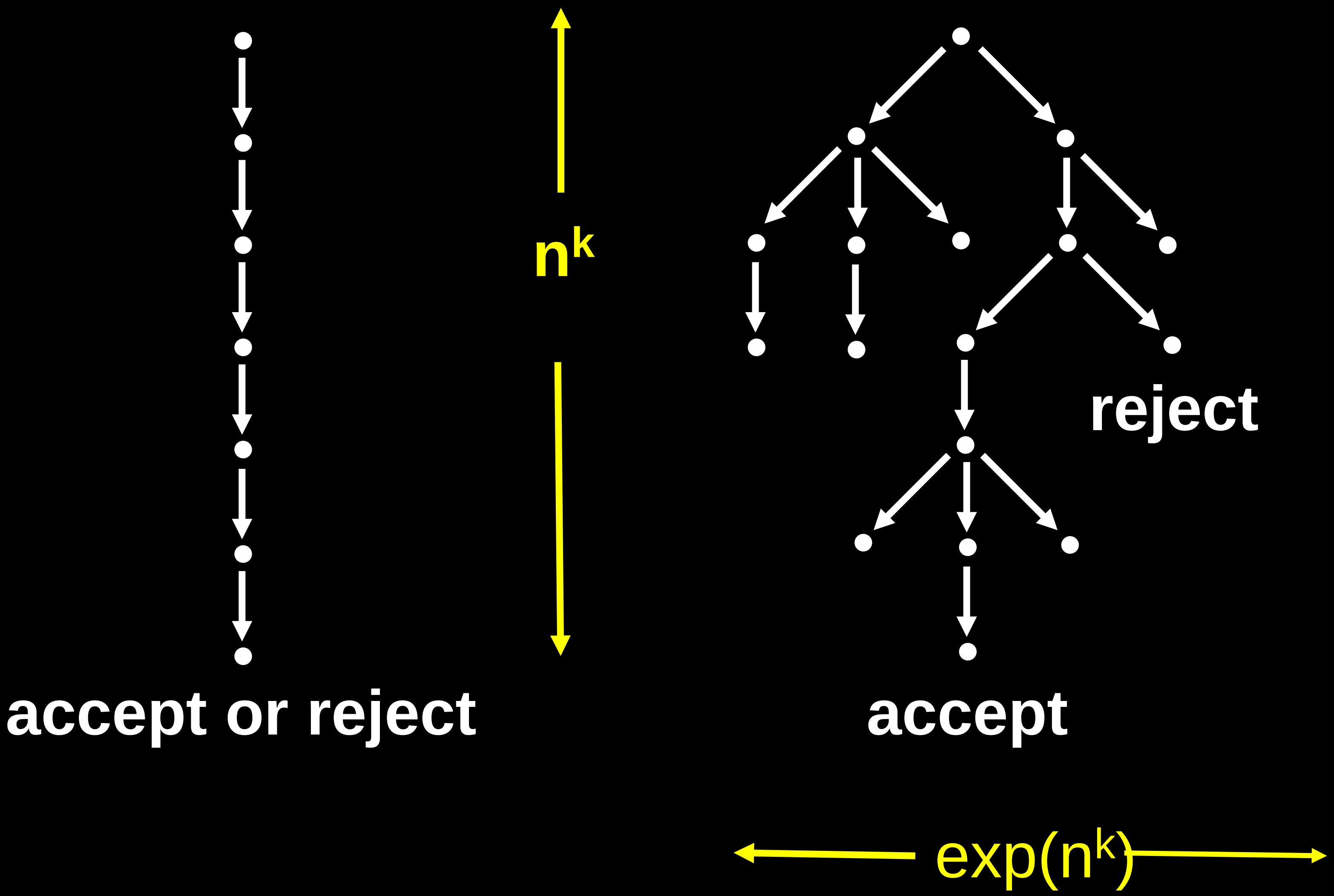
ϕ will simulate the NP machine N for A on w.

So proof will also show: 3-SAT is NP-Complete



Deterministic Computation

Non-Deterministic Computation



accept or reject

n^k

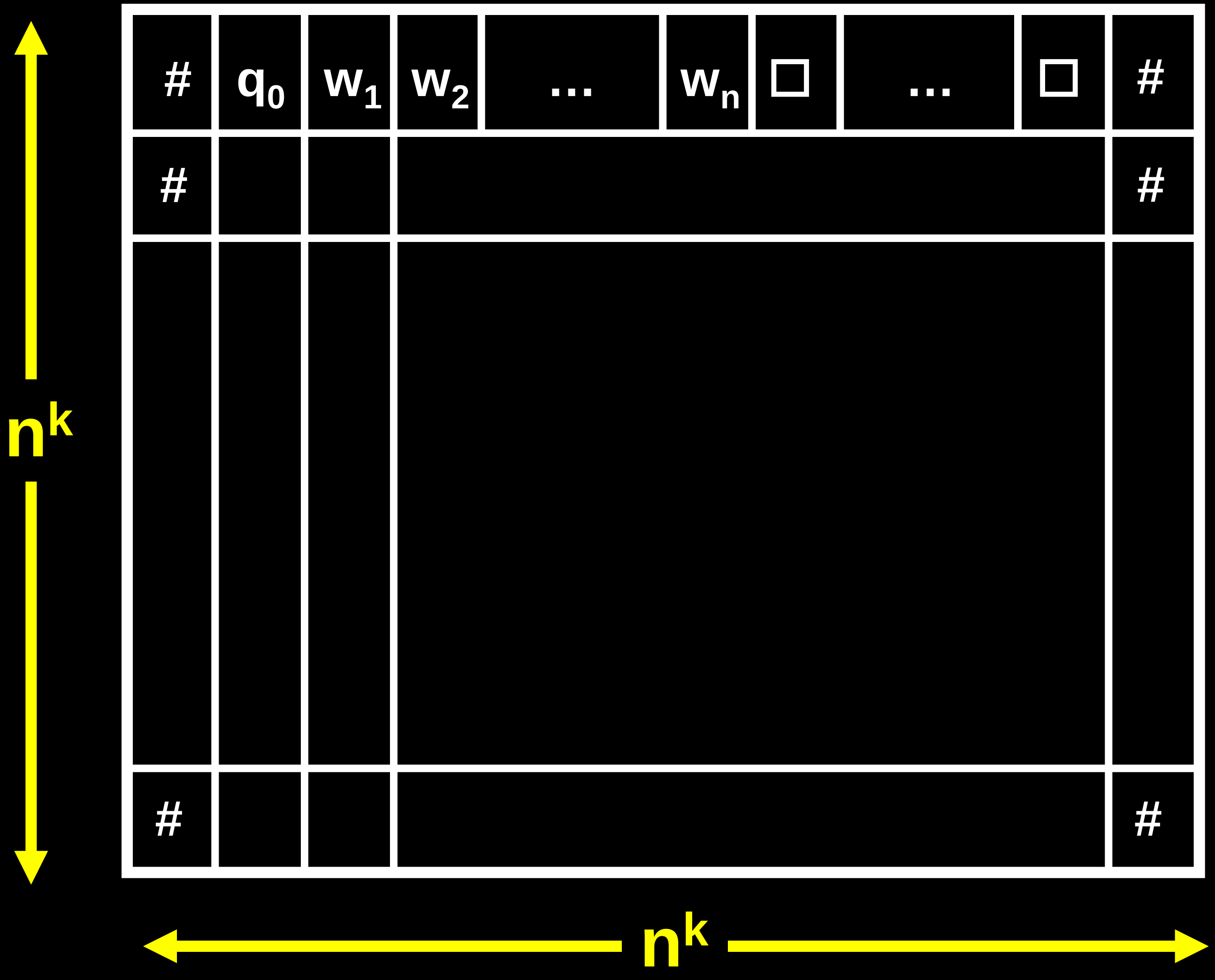
reject

accept

$\exp(n^k)$

Suppose A ∈ NTIME(n^k) and let N be an NP machine for A.

A **tableau for N on w** is an n^k × n^k table whose rows are the configurations of *some* possible computation of N on input w.



#	q_0	w_1	w_2	…	w_n	□	…	□	#
#									#
#									#

n^k

n^k

A tableau is accepting if any row of the tableau is an accepting configuration

Determining whether N accepts w is equivalent to determining whether there is an accepting tableau for N on w

A tableau is accepting if any row of the tableau is an accepting configuration

Determining whether N accepts w is equivalent to determining whether there is an accepting tableau for N on w

Given w, our 3cnf-formula ϕ will describe a *generic* tableau for N on w (in fact, essentially *generic* for N on any string w of length n).

The 3cnf formula ϕ will be satisfiable *if and only if* there is an accepting tableau for N on w.

VARIABLES of ϕ

Let $C = Q \cup \Gamma \cup \{ \# \}$

Each of the $(n^k)^2$ entries of a tableau is a cell

cell[i,j] = the cell at row i and column j

For each i and j ($1 \leq i, j \leq n^k$) and for each $s \in C$ we have a variable $x_{i,j,s}$

# variables = $|C|n^{2k}$, ie $O(n^{2k})$, since $|C|$ only depends on N

VARIABLES of ϕ

Let $C = Q \cup \Gamma \cup \{ \# \}$

Each of the $(n^k)^2$ entries of a tableau is a cell

cell[i,j] = the cell at row i and column j

For each i and j ($1 \leq i, j \leq n^k$) and for each $s \in C$ we have a variable $x_{i,j,s}$

# variables = $|C|n^{2k}$, ie $O(n^{2k})$, since $|C|$ only depends on N

These are the variables of ϕ and represent the contents of the cells

We will have: $x_{i,j,s} = 1 \Leftrightarrow \text{cell}[i,j] = s$

$$x_{i,j,s} = 1$$

means

$$\text{cell}[\ i,\ j\ ] = s$$

We now design ϕ so that a satisfying assignment to the variables $x_{i,j,s}$ corresponds to an accepting tableau for N on w

The formula ϕ will be the **AND** of four parts:

$$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{accept} \wedge \phi_{move}$$

We now design ϕ so that a satisfying assignment to the variables $x_{i,j,s}$ corresponds to an accepting tableau for **N** on **w**

The formula ϕ will be the **AND** of four parts:

$$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{accept} \wedge \phi_{move}$$

ϕ_{cell} ensures that for each i,j, exactly one $x_{i,j,s} = 1$

ϕ_{start} ensures that the first row of the table is the *starting (initial)* configuration of **N** on **w**

ϕ_{accept} ensures* that an accepting configuration occurs somewhere in the table

ϕ_{move} ensures* that every row is a configuration that legally follows from the previous config

*if the other components of ϕ hold

ϕ_{cell} ensures that for each i,j, exactly one $x_{i,j,s} = 1$

$$\phi_{cell} = \bigwedge_{1 \le i,\, j \le n^k} \left[\left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s,t \in C \\ s \ne t}} (\neg x_{i,j,s} \vee \neg x_{i,j,t}) \right) \right]$$

at least one variable is turned on

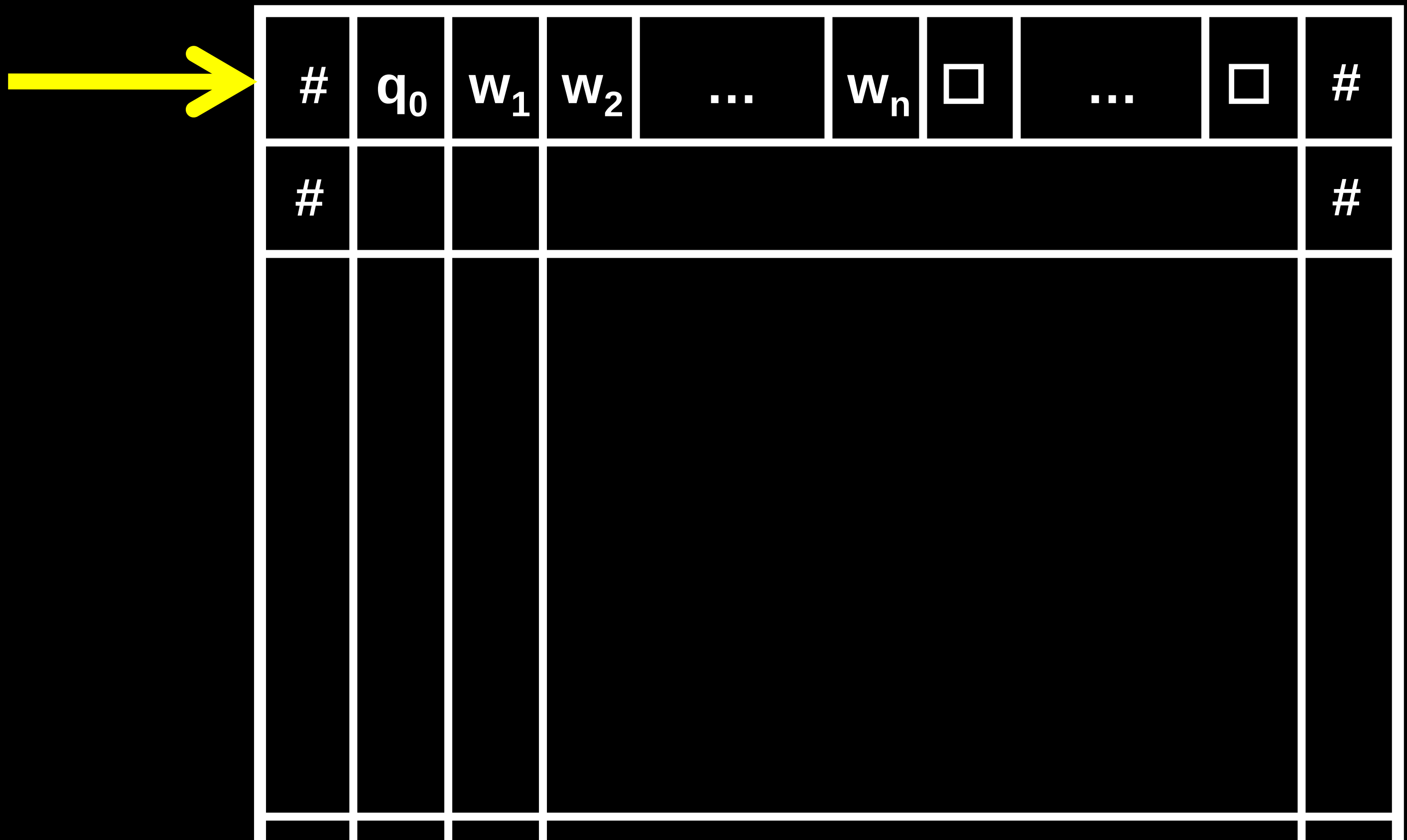
at most one variable is turned on

ϕ_{cell} ensures that for each i,j, exactly one $x_{i,j,s} = 1$

$$\phi_{cell} = \bigwedge_{1 \le i,\, j \le n^k} \left[\underbrace{\left(\bigvee_{s \in C} x_{i,j,s} \right)}_{\text{at least one variable is turned on}} \wedge \underbrace{\left(\bigwedge_{\substack{s,t \in C \\ s \neq t}} (\neg x_{i,j,s} \vee \neg x_{i,j,t}) \right)}_{\text{at most one variable is turned on}} \right]$$

Thus, ϕ_{cell} is satisfiable
(ie, there exist assignment to the variables s.t. ϕ_{cell} evaluates to 1)
$\Leftrightarrow$
each cell in the **tableau** has exactly one symbol (from C.)

$$\begin{aligned}\phi_{start} = \; & x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \\ & x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge \\ & x_{1,n+3,\square} \wedge \cdots \wedge x_{1,n^k-1,\square} \wedge x_{1,n^k,\#}\end{aligned}$$



#	q_0	w_1	w_2	…	w_n	□	…	□	#
#									#

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\square} \wedge \cdots \wedge x_{1,n^k-1,\square} \wedge x_{1,n^k,\#}$$

Thus, ϕ_{start} is satisfiable $\Leftrightarrow$ the first row of the tableau represents the start configuration for N on input w

$$\phi_{accept} = \bigvee_{1 \leq i, j \leq n^k} x_{i,j,q_{accept}}$$

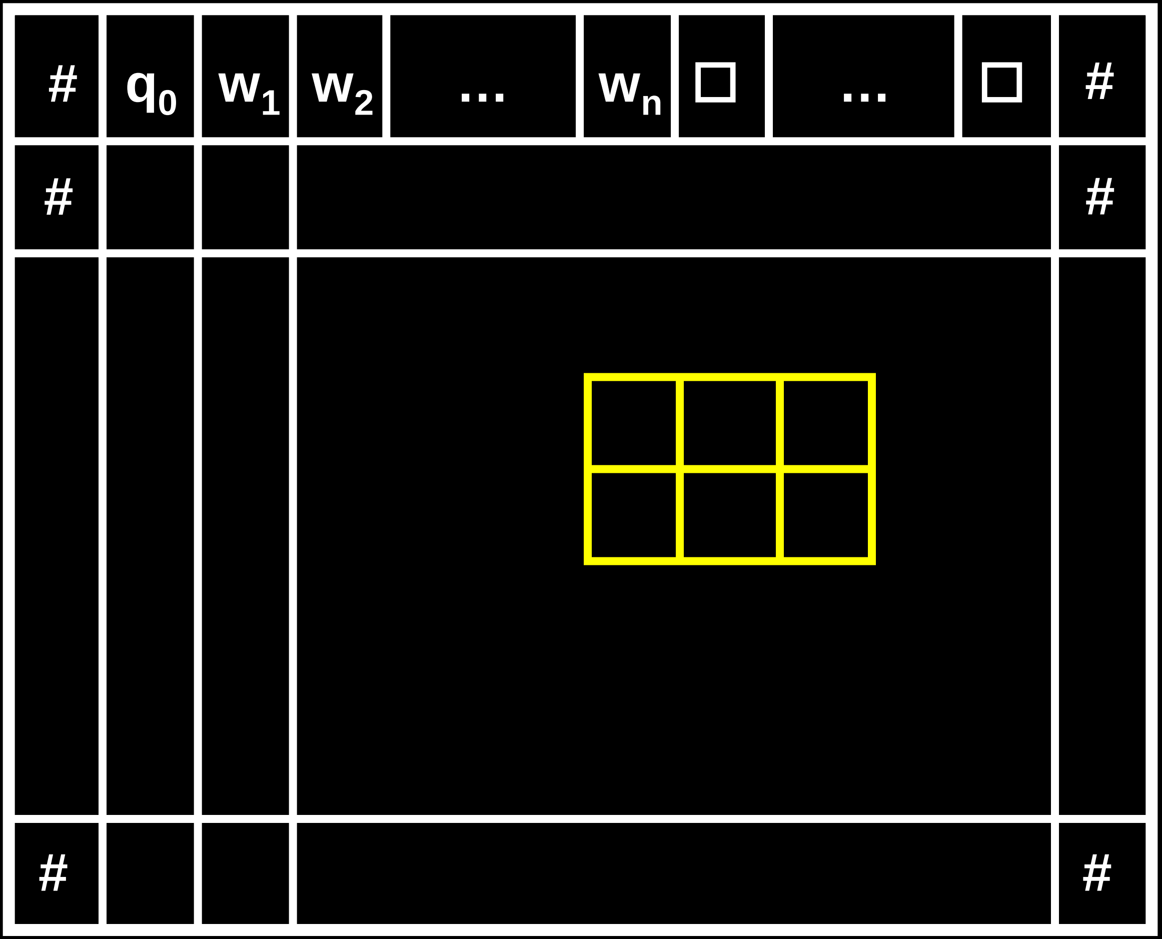
Thus, ϕ_{accept} is satisfiable ⇔
at least one cell in the tableau has the symbol q_{accept}.

ϕ_{move} ensures that every row is a configuration that legally follows from the previous

It works by ensuring that each 2 × 3 “window” of cells is legal (Does not violate N’s rules)

ϕ_{move} ensures that every row is a configuration that legally follows from the previous

It works by ensuring that each 2 × 3 "window" of cells is legal (Does not violate N's rules)



If $\delta(q_1,a) = \{(q_1,b,R)\}$ and $\delta(q_1,b) = \{(q_2,c,L), (q_2,a,R)\}$ which of the following windows are legal:

a	q_1	b
q_2	a	c

a	q_1	b
q_1	a	a

a	a	q_1
a	a	b

#	b	a
#	b	a

a	b	a
a	b	q_2

b	q_1	b
q_2	b	q_2

a	b	a
a	a	a

a	q_1	b
a	a	q_2

b	b	b
c	b	b

If $\delta(q_1,a) = \{(q_1,b,R)\}$ and $\delta(q_1,b) = \{(q_2,c,L), (q_2,a,R)\}$ which of the following windows are legal:

a	q_1	b
q_2	a	c

🚫

a	q_1	b
q_1	a	a

a	a	q_1
a	a	b

#	b	a
#	b	a

a	b	a
a	b	q_2

🚫

b	q_1	b
q_2	b	q_2

🚫

a	b	a
a	a	a

a	q_1	b
a	a	q_2

b	b	b
c	b	b

CLAIM:

If

- **the top row of the table is the start configuration, and**
- **and every window is legal,**

Then

each row of the table is a configuration that legally follows the preceding one.

CLAIM:

If

- the top row of the table is the start configuration, and
- and every window is legal,

Then

each row of the table is a configuration that legally follows the preceding one.

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

CLAIM:

If

- **the top row of the table is the start configuration, and**
- **and every window is legal,**

Then

each row of the table is a configuration that legally follows the preceding one.

	a	
	a	

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol

CLAIM:

If

- **the top row of the table is the start configuration, and**
- **and every window is legal,**

Then

each row of the table is a configuration that legally follows the preceding one.

	a	
	a	

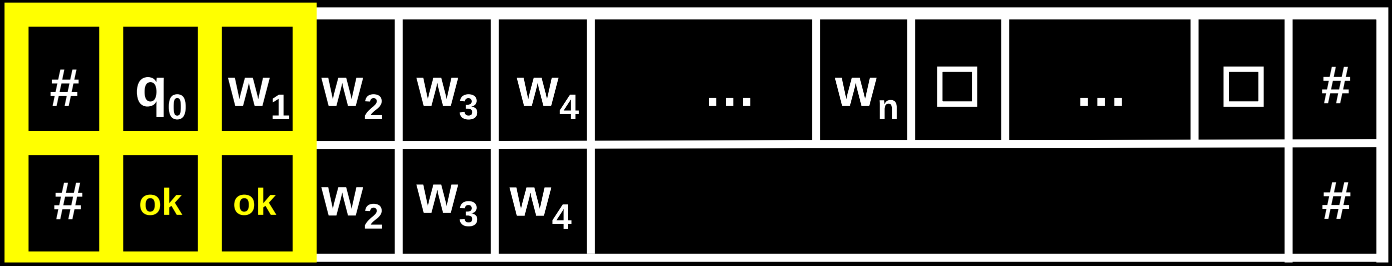
	q	

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol

Case 2. center cell of window is a state symbol

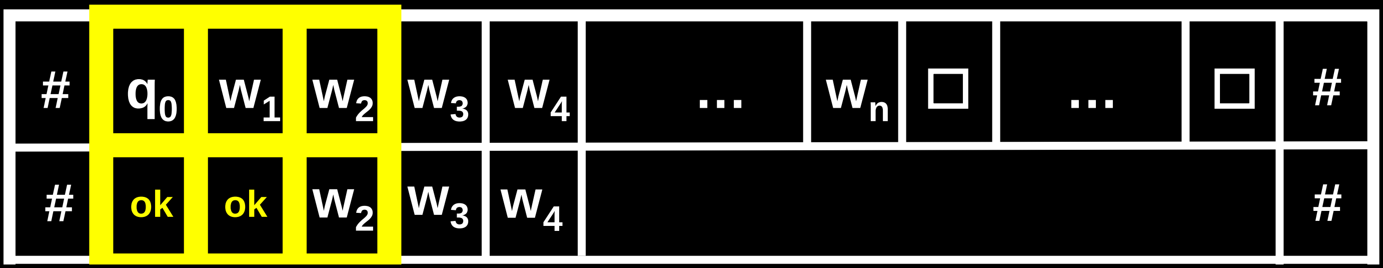


#	q_0	w_1	w_2	w_3	w_4	…	w_n	□	…	□	#
#	ok	ok	w_2	w_3	w_4						#

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol
Case 2. center cell of window is a state symbol

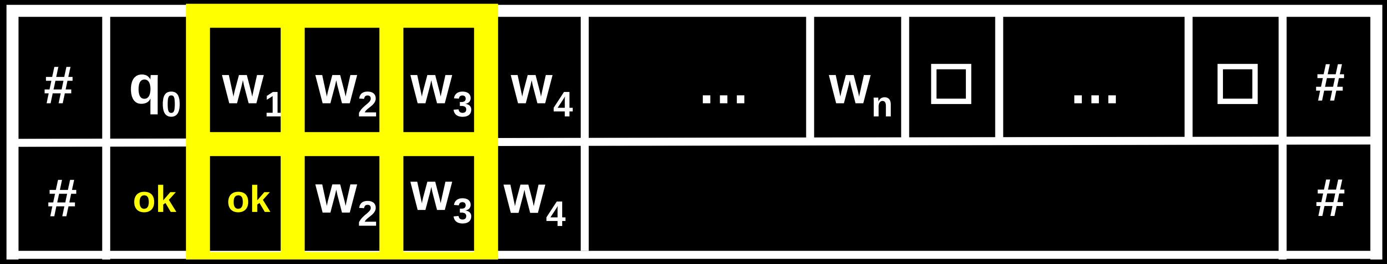


#	q_0	w_1	w_2	w_3	w_4	…	w_n	□	…	□	#
#	ok	ok	w_2	w_3	w_4						#

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol
Case 2. center cell of window is a state symbol



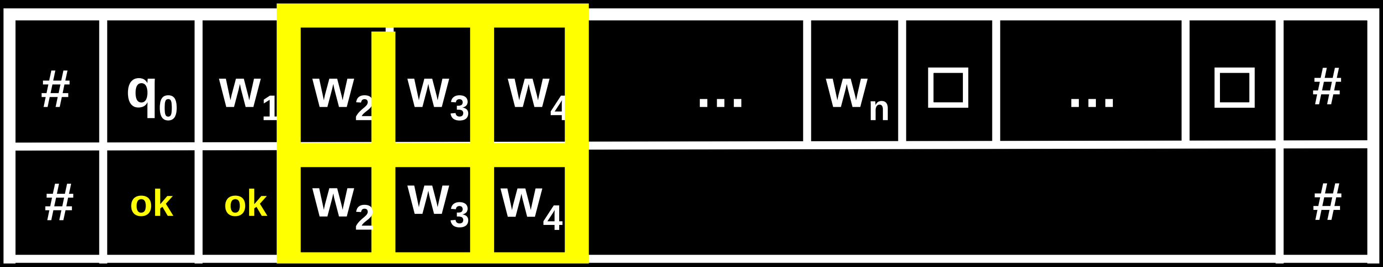
#	q_0	w_1	w_2	w_3	w_4	…	w_n	□	…	□	#
#	ok	ok	w_2	w_3	w_4						#

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol

Case 2. center cell of window is a state symbol

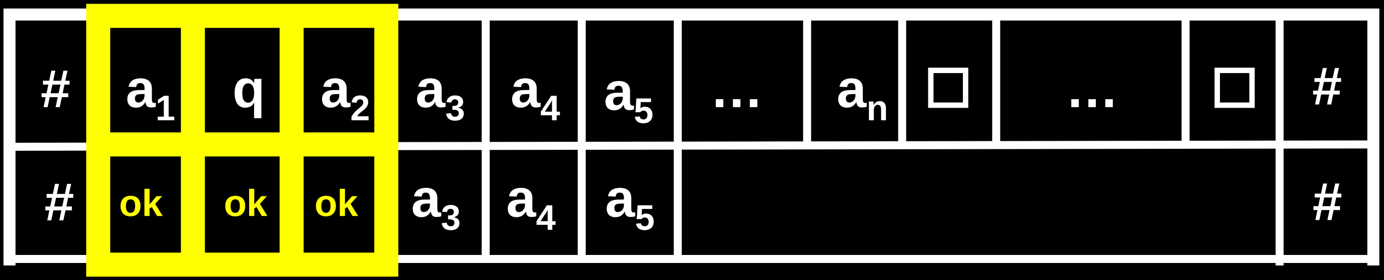


#	q_0	w_1	w_2	w_3	w_4	…	w_n	□	…	□	#
#	ok	ok	w_2	w_3	w_4						#

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol
Case 2. center cell of window is a state symbol

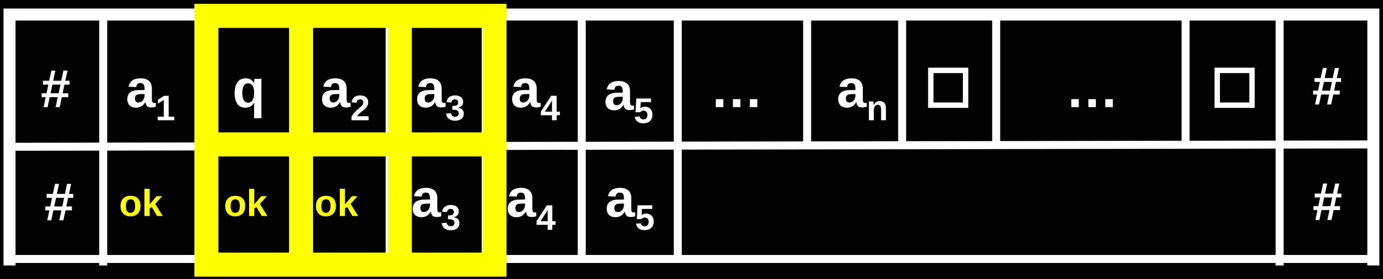


#	a_1	q	a_2	a_3	a_4	a_5	…	a_n	□	…	□	#
#	ok	ok	ok	a_3	a_4	a_5						#

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol
Case 2. center cell of window is a state symbol

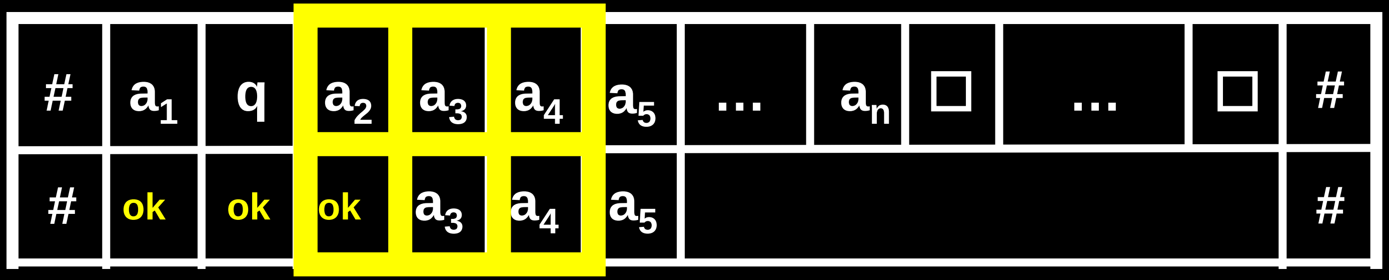


#	a_1	q	a_2	a_3	a_4	a_5	…	a_n	□	…	□	#
#	ok	ok	ok	a_3	a_4	a_5						#

Proof:

In upper configuration, every cell that doesn't contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol
Case 2. center cell of window is a state symbol



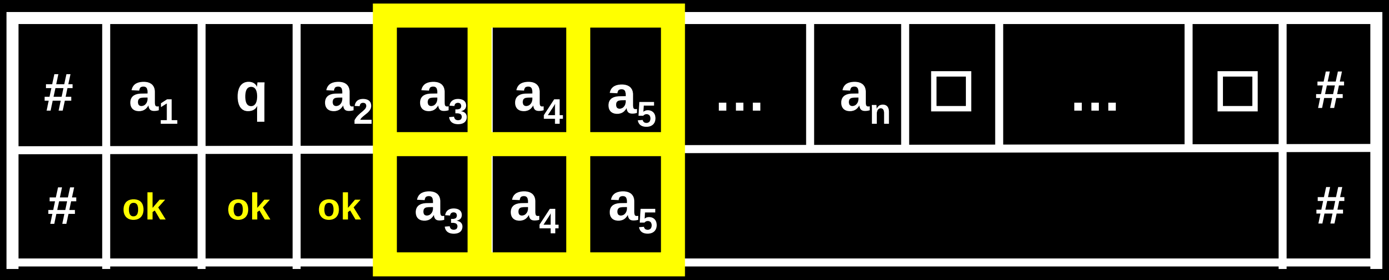
#	a_1	q	a_2	a_3	a_4	a_5	…	a_n	□	…	□	#
#	ok	ok	ok	a_3	a_4	a_5						#

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

Case 1. center cell of window is a non-state symbol and not adjacent to a state symbol

Case 2. center cell of window is a state symbol



#	a_1	q	a_2	a_3	a_4	a_5	…	a_n	□	…	□	#
#	ok	ok	ok	a_3	a_4	a_5						#

Proof:

In upper configuration, every cell that doesn’t contain the boundary symbol #, is the center top cell of a window.

So the lower configuration follows from the upper!!!

	col. j-1	col. j	col. j+1
row i	**(i,j-1)** $\mathbf{a_1}$	**(i,j)** $\mathbf{a_2}$	**(i,j+1)** $\mathbf{a_3}$
row i+1	**(i+1,j-1)** $\mathbf{a_4}$	**(i+1,j)** $\mathbf{a_5}$	**(i+1,j+ 1)** $\mathbf{a_6}$

The (i,j) Window

$$\phi_{move} = \bigwedge_{1 \leq i, j \leq n^k} (\text{ the } (i, j) \text{ window is legal })$$

the (i, j) window is legal =

$$\bigvee_{\substack{a_1, \ldots, a_6 \\ \text{is a legal window}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j,+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

This is a disjunct over all ($\leq |C|^6$) legal sequences ($a_1, \ldots, a_6$).

$$\phi_{\text{move}} = \bigwedge_{1 \le i, j \le n^k} (\text{ the } (i, j) \text{ window is legal })$$

the (i, j) window is legal =

$$\bigvee_{\substack{a_1, \ldots, a_6 \\ \text{is a legal window}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j,+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

This is a disjunct over all (≤ $|C|^6$) legal sequences ($a_1, \ldots, a_6$).

This disjunct is satisfiable

⇔

There is *some* assignment to the cells (ie variables) in the window (i,j) that makes the window legal

$$\phi_{\text{move}} = \bigwedge_{1 \le i, j \le n^k} (\text{ the } (i, j) \text{ window is legal })$$

the (i, j) window is legal =

$$\bigvee_{\substack{a_1, \dots, a_6 \\ \text{is a legal window}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j,+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

This is a disjunct over all (≤ $|C|^6$) legal sequences ($a_1, \dots, a_6$).

So ϕ_{move} is satisfiable

⇔

There is *some* assignment to each of the variables that makes *every* window legal.

$$\phi_{\text{move}} = \bigwedge_{1 \le i, j \le n^k} (\text{ the } (i, j) \text{ window is legal })$$

the (i, j) window is legal =

$$\bigvee_{\substack{a_1, \ldots, a_6 \\ \text{is a legal window}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j,+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

This is a disjunct over all ($\le |C|^6$) legal sequences ($a_1, \ldots, a_6$).

Can re-write as equivalent conjunct:

$$\phi_{\text{move}} = \bigwedge_{1 \le i,\, j \le n^k} (\text{ the } (i, j) \text{ window is legal })$$

the (i, j) window is legal =

$$\bigvee_{\substack{a_1, \ldots, a_6 \\ \text{is a legal window}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j,+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6})$$

This is a disjunct over all ($\le |C|^6$) legal sequences ($a_1, \ldots, a_6$).

Can re-write as equivalent conjunct:

$$\equiv \bigwedge_{\substack{a_1, \ldots, a_6 \\ \text{ISN'T a legal window}}} (\bar{x}_{i,j-1,a_1} \vee \bar{x}_{i,j,a_2} \vee \bar{x}_{i,j,+1,a_3} \vee \bar{x}_{i+1,j-1,a_4} \vee \bar{x}_{i+1,j,a_5} \vee \bar{x}_{+1,j+1,a_6})$$

This is a conjunct over all ($\le |C|^6$) illegal sequences ($a_1, \ldots, a_6$).

$$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{accept} \wedge \phi_{move}$$

ϕ is satisfiable (ie, there is some assignment to each of the varialbes s.t. ϕ evaluates to 1)

⇔

there is some assignment to each of the variables s.t. ϕ_{cell} and ϕ_{start} and ϕ_{accept} and ϕ_{move} each evaluates to 1

⇔

There is some assignment of symbols to cells in the tableau such that:

- The first row of the tableau is a start configuration and
- Every row of the tableau is a configuration that follows from the preceding by the rules of N and
- One row is an accepting configuration

,⇔

There is some accepting computation for N with input w

$$\phi = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{accept}} \wedge \phi_{\text{move}}$$

WHAT'S THE LENGTH OF ϕ ?

$$\phi = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{accept}} \wedge \phi_{\text{move}}$$

$$\phi_{\text{cell}} = \bigwedge_{1 \leq i,\, j \leq n^k} \left[\left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s,t \in C \\ s \neq t}} (\neg x_{i,j,s} \vee \neg x_{i,j,t}) \right) \right]$$

$O(n^{2k})$ clauses

$$\text{Length}(\phi_{\text{cell}}) = O(n^{2k})\, \underbrace{O(\log(n))}_{\text{length(indices)}} = O(n^{2k} \log n)$$

$$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{accept} \wedge \phi_{move}$$

$$\begin{aligned}\phi_{start} = \; & x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \\ & x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge \\ & x_{1,n+3,\square} \wedge \cdots \wedge x_{1,n^k-1,\square} \wedge x_{1,n^k,\#}\end{aligned}$$

$$O(n^k)$$

$$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{accept} \wedge \phi_{move}$$

$$\phi_{accept} = \bigvee_{1 \leq i,\, j \leq n^k} x_{i,j,q_{accept}}$$

$$O(n^{2k})$$

$$\phi_{\text{move}} = \bigwedge_{1 \le i,\, j \le n^k} (\text{ the } (i, j) \text{ window is legal })$$

the (i, j) window is legal =

$$\bigwedge_{a_1, \ldots, a_6} (\overline{x}_{i,j-1,a_1} \vee \overline{x}_{i,j,a_2} \vee \overline{x}_{i,j,+1,a_3} \vee \overline{x}_{i+1,j-1,a_4} \vee \overline{x}_{i+1,j,a_5} \vee \overline{x}_{i+1,j+1,a_6})$$

ISN’T a legal window

This is a conjunct over all (≤ $|C|^6$) illegal sequences (a_1, …, a_6).

$$O(n^{2k})$$

Theorem (Cook-Levin): SAT is NP-complete

Corollary: SAT $\in$ P if and only if P = NP

Theorem (Cook-Levin): 3SAT is NP-complete

Corollary: 3SAT $\in$ P if and only if P = NP

3-SAT?

How do we convert the whole thing into a 3-cnf formula?

Everything was an AND of ORs
We just need to make those ORs with 3 literals

If a clause has less than three variables:

$\mathbf{a \equiv (a \lor a \lor a), \ (a \lor b) \equiv (a \lor b \lor b)}$

3-SAT?

How do we convert the whole thing into a 3-cnf formula?

Everything was an AND of ORs

We just need to make those ORs with 3 literals

If a clause has less than three variables:

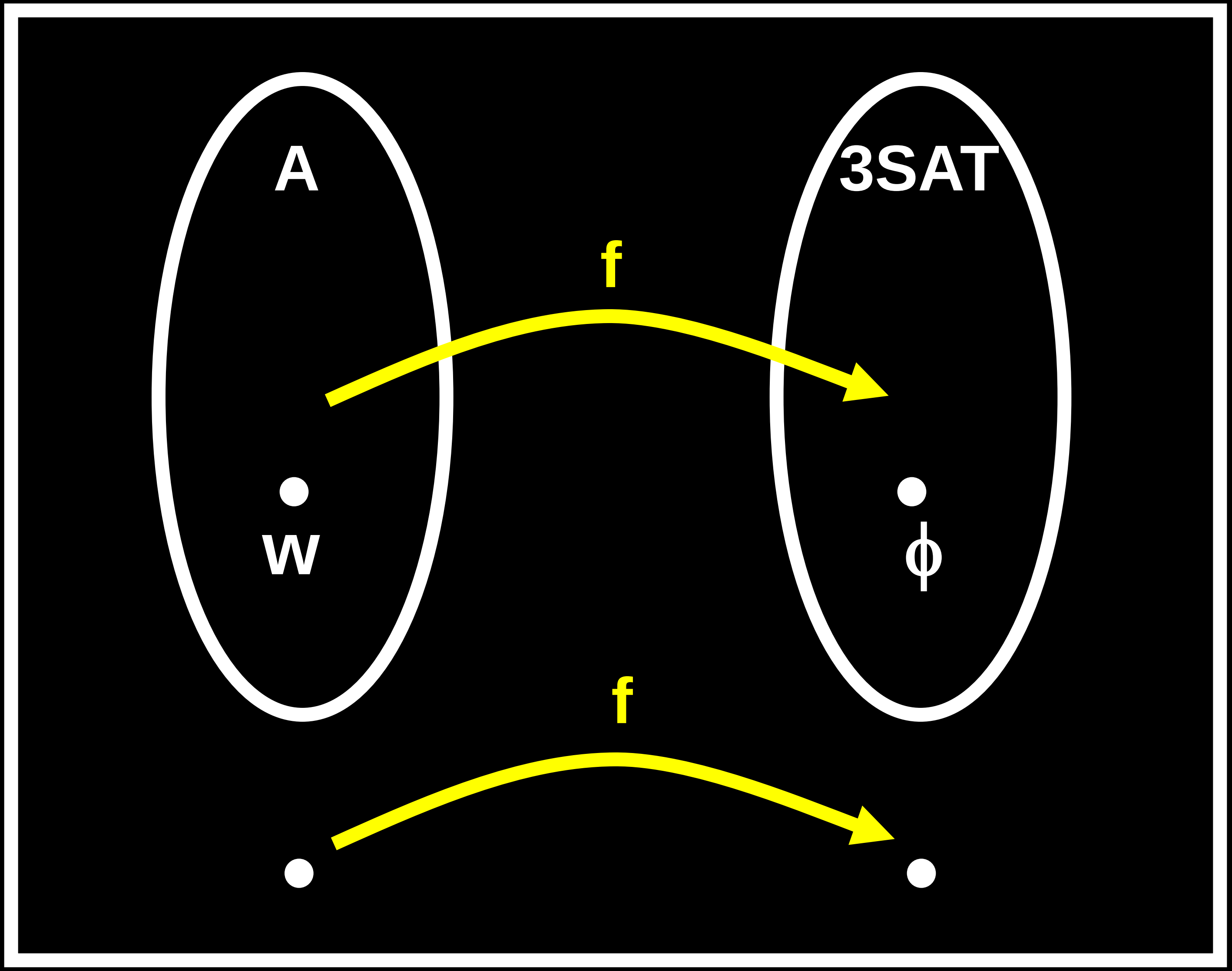
$$\mathbf{a \equiv (a \lor a \lor a), \quad (a \lor b) \equiv (a \lor b \lor b)}$$

If a clause has more than three variables:

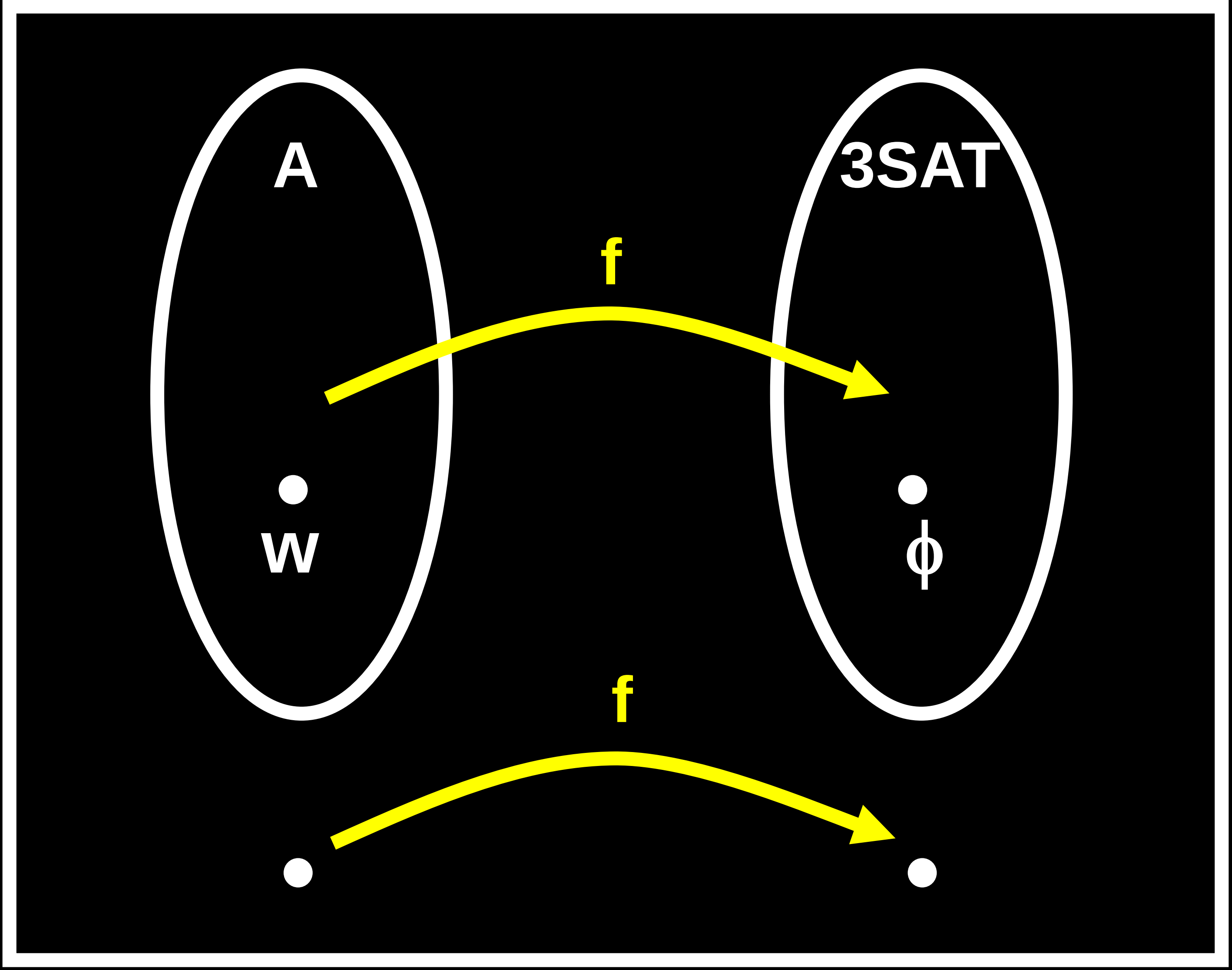
$$\mathbf{(a \lor b \lor c \lor d) \equiv (a \lor b \lor z) \land (\neg z \lor c \lor d)}$$

$$\mathbf{(a_1 \lor a_2 \lor \ldots \lor a_t) \equiv}$$

$$\mathbf{(a_1 \lor a_2 \lor z_1) \land (\neg z_1 \lor a_3 \lor z_2) \land \ldots (\neg z_{t-3} \lor a_{t-1} \lor z_t)}$$

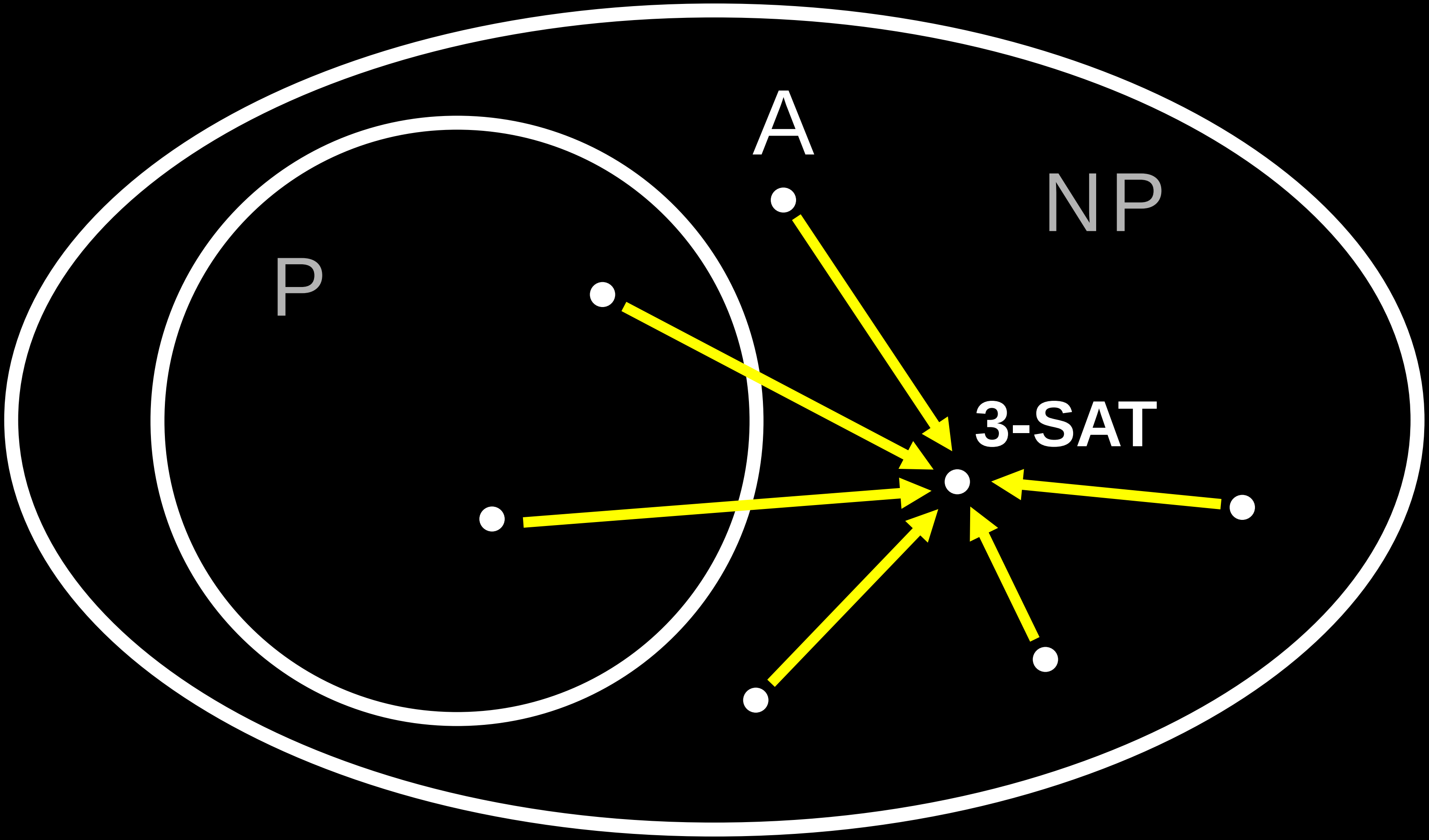


Given A in NP. The reduction f turned a string w into a 3-cnf formula ϕ such that: w ∈ A ⇔ ϕ ∈ 3SAT.



The reduction f is poly time. WHY?

3-SAT is NP-Complete



Theorem (Cook-Levin): 3SAT is NP-complete

Corollary: 3SAT $\in$ P if and only if P = NP