TQBF $\in$ PSPACE - Complete

PB. TQBF $\in$ PSPACE (maybe proof of there's time, recursion)

$\text{NP} \leq_{p} \text{TQBF}$

Let $M$ be a poly-space TM deciding $L$

WLOG, $M$ has a unique accepting configuration

Draw $M$ has an accepting configuration (Cook-Levin)

with 1 row and $\text{poly}(m) + 3$ columns.

Let

```
1st row: #a_0 #a_1 #a_2 #a_3 w_1 w_2 w_3 w_4
```

Last row of $T$:

```
#a_0 #a_1 #a_2 #a_3
```

Let

```
#w_1 b_2 #1 #b_0 c
```

We can make the TM to erase # before accepting.

So we code each formula has one variable (one for each cell)

$\Phi = \text{Find } \Phi_{\text{start}} \land \Phi_{\text{more}} \land \Phi_{\text{accept}}$

- $\Phi_{\text{start}}$ is a valid configuration such that:
  - a valid test
  - last row of $T$ contains $\#a_0 a_1 a_2 a_3$ [1 column]
    - top row of $T$ contains $\#a_0 a_1 a_2 a_3$
    - 1 row 1 column: this simulates $M$'s start and ends in $\text{NP}$

How can we simulate $M$'s configuration to copy $c$ in $T$ steps?

Let $T$ be a quasilinear formula formula that is true if $M$'s start configuration to copy $c$ in $T$ steps.
Compute \( \psi(C_1, C_2, \ldots) \)

\[ t_6 + c_0 = c_1 = c_2, \text{ouput} \]

\[ t_6 + 1 \]

let \( a_1, \ldots, a_d \) be elements in \( K \) that correspond to \( C_1 \) and \( C_2 \)
\[ c_1 = \#a_1 + a_2 + \ldots + a_d \]

let \( a_1, a_2, a_3 \) be elements in \( K \) that correspond to \( C_2 \)
\[ \text{output } \Phi \text{move}(a_1, a_2, a_3) \]
where \( \Phi \text{move} \) clauses are those that follow from using \( \Phi \text{move} \) clauses.

\[ t_6 + 2 \]
\[ \text{output } \Phi \]

\[ \exists x_1 \ldots \exists x_d \left( \Phi \text{move}(x_1, \ldots, x_d) \right) \]
\[ \land \Phi \text{move}(a_1, a_2, a_3) \]
\[ \land \Phi \text{move}(x_1, \ldots, x_d) \]

\[ \text{No output } \exists x_1 \ldots \exists x_d s.t. \forall y_1, \ldots, y_d \exists z_1, \ldots, z_d \]
\[ \Phi \text{cell}(x_1, \ldots, x_d) \land \left( (y_1 \neq a_1) \land (y_2 \neq a_2) \right) \land \Phi \text{move}(y_1, z_1) \]

\[ \Phi \text{output} \]

must be the same number and close to the \( C_1 \) and \( C_2 \) correspondingly.

Booleans \( x_0 \exists x \exists x \exists y \exists \bar{y} \exists \bar{z} \exists \bar{w} \exists \bar{u} \exists \bar{v} \exists \bar{w} \Phi \text{cell}(x, \ldots, x) \land \Phi \text{move}(y, z) \)
Compute \( \Psi_i(C_1, C_2, t) \)

\[
T_{i+1} > 2 \text{ \textcolor{red}{WHY?}} \text{ \textcolor{red}{WHY?}} \text{ \textcolor{red}{WHY?}}
\]

\[
2t = \exists x, \forall (y, z) \in \{\text{pred}(x), \text{pred}(x, t)\} \forall \text{cell}(x) \land \text{compute } \Psi_i(y, z, t)
\]

\[
\text{produce by}
\]

\[
\text{compute all}
\]

Reduce \( \text{chain} \)

On input \( w \), compute \( a \), a \text{back assignment encoding } T^{th} \text{ clause tableau}

- \text{Back assignment encoding the last row of tableau}

Output \( \rightarrow \text{Compute } \Psi_i(a, b, t) \)

Runtime \( t = \text{poly}(w) + \text{runtime } T/2 \)

- \( \text{poly}(w) \cdot \log t \)
- \( \text{poly} \cdot 2(w, l) \)

\( \text{O.E. PSPACE COMPLETE} \)

**SPACE** and other Space Complexity Classes

\( I \) = Logarithmic space. How is that possible?

Consider TMs that have 2 tapes

- Tape 1: Read only input tape

- Tape 2: Work tape \( \rightarrow \text{new} \text{ Turing machine limited to 2 tape} \)

\( I \) = \{ \# \mid \text{A can be decided by a log-space TM as defined above} \}

\( \# = O(\log n) \text{ space on work tape} \)
Example: \( A = \{ a^n b^n c^n \} \) requires logarithmic space in a pushdown.

Is NL decided by a nondeterministic logspace TM?

PATH = \( (G_1, t_1, t_2) \) from \( s \) to \( t \)

Undirected G encoded as a list of vertex + edge.

How would you decide? 

Labeled \( V_1 \rightarrow V_2 \rightarrow V_3 \rightarrow \ldots \rightarrow V_3 \rightarrow V_2 \rightarrow V_1 \)

Write tape \( V_2 \rightarrow 0 \rightarrow n \rightarrow V_1 \)

can move until you see an edge on write-only tape, underenumerally advance "take" edge, incrementing your counter until you reach the target vertex.

\( PATH \) is \( NL \)-complete.

\( NL \)-completeness is defined differently:

- Logspace, nondeterministic 3-tape TM, \( \delta \) read-only input tape
- Logspace workspace
- Write only output tape

\( A \leq^L_B \) \( \text{if logspace \ \exists \text{TM}(f) \ s.t. } f(x) \in A \leftrightarrow f(x) \in B \)

Def \( A \) is \( NL \)-complete if

1. \( A \in NL \)
2. \( \forall B \in NL, B \leq_A A \)

Then \( 1.B \) \( A \text{ is } NL \)-complete, \( + \text{ is } NL \), \( 2 \in NL \).

Then \( \emptyset \) \( NL \)-complete.

Then \( NL = coNL \).