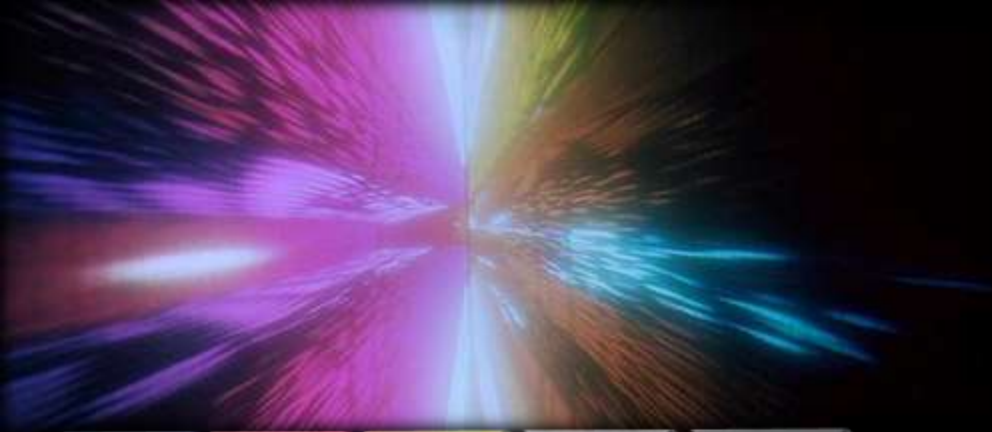


# SPACE

# COMPLEXITY

## Goal:

- Explore space complexity

## Plan:

- Low space classes: L, NL
- Savitch's Theorem
- Immerman's Theorem
- TQBF

# Space-Complexity

## Definition:

- Let $t:N \rightarrow N$ be a complexity function

## Deterministic space:

$$SPACE[t(n)] \cong \{L \mid L \text{ decided by } O(t(n))\text{-space } deterministic \text{ TM}\}$$

## Nondeterministic space:

$$NSPACE[t(n)] \cong \{L \mid L \text{ decided by } O(t(n))\text{-space non} deterministic \text{ TM}\}$$

?: the input takes n cells: how can a TM use only logn space?

## Det. Log space:

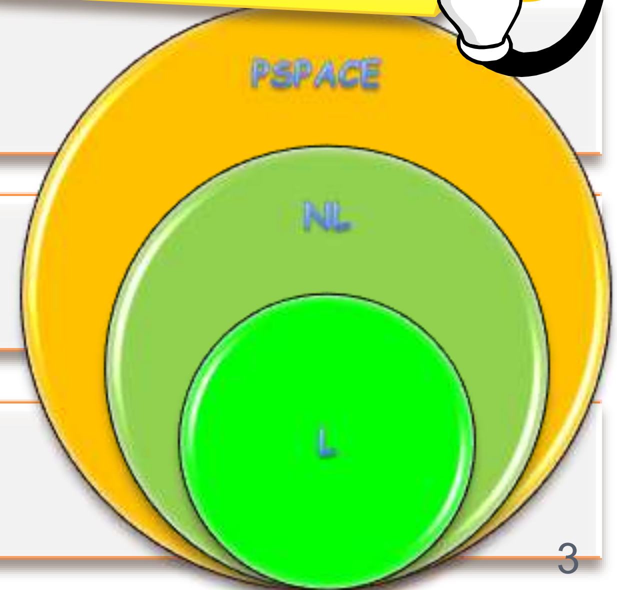
$$L \equiv SPACE[\log(n)]$$

## Nondet. Log space:

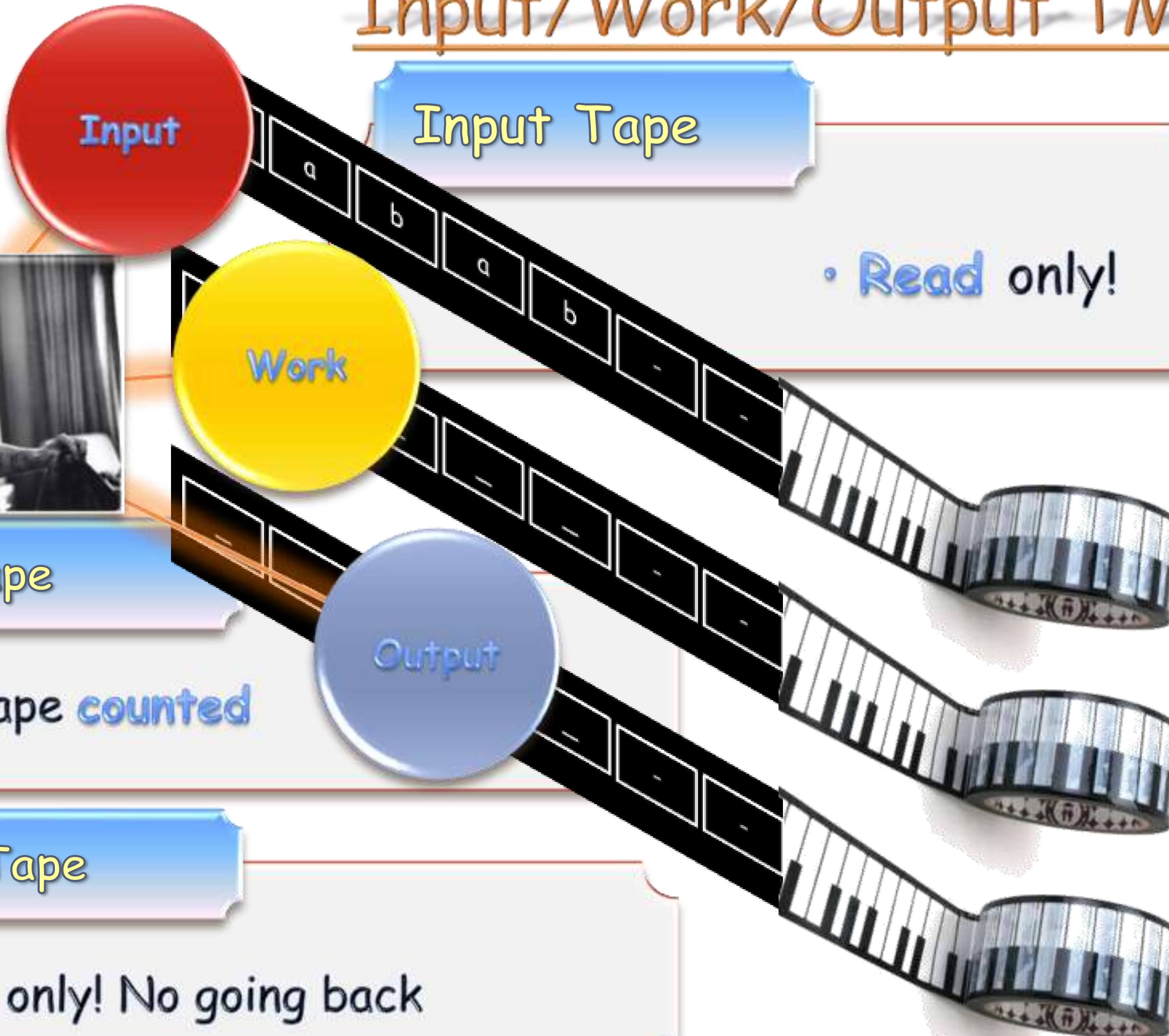
$$NL \equiv NSPACE[\log(n)]$$

## Det polynomial space:

$$PSPACE \equiv \bigcup_k SPACE[n^k]$$



# Input/Work/Output TM




## Input Tape

- **Read** only!

## Work Tape

- Only tape **counted**

## Output Tape

- **Write** only! No going back

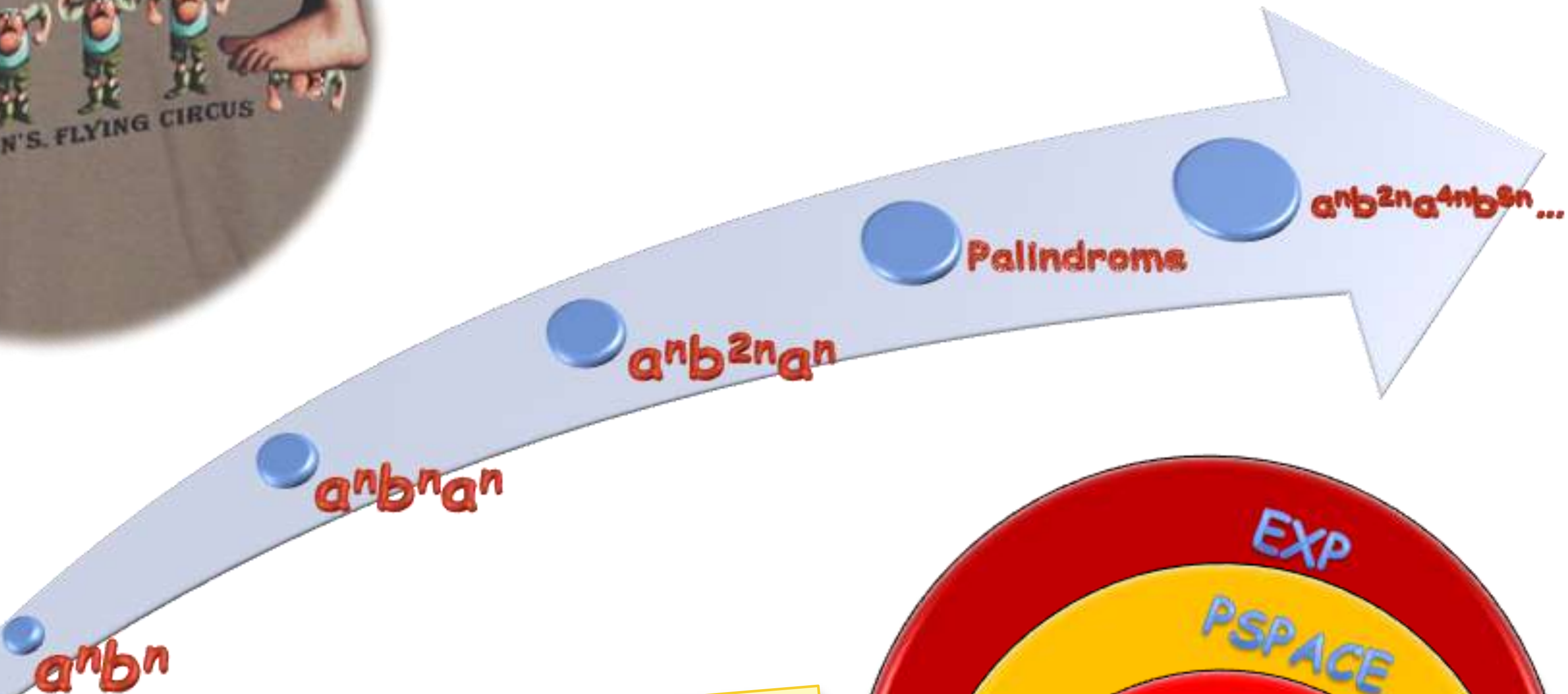
# Configurations

How many distinct configurations may a TM with input-size N and work-tape of size S have?


What about output?

| content of the input tape | Head position Input | content of the work tape | Head position Work | the machine's state |
|---|---|---|---|---|
| $\lvert\Sigma\rvert^N \times$ | $N \times$ | $\lvert\Gamma\rvert^S \times$ | $S \times$ | $\lvert Q\rvert$ |

# Brain Hurts


**Find**
A problem in NL
Not known to be in L

# Log-space Reductions

A is <u>log-space reducible</u> to B (denoted $A \leq_L B$)

If there exists a log-space-computable function $f:\Sigma^* \to \Sigma^*$

i.e., ∃ log-space TM that outputs f(w) on input w

s.t. for every w

$w \in A \Leftrightarrow f(w) \in B$

f is a log-space <u>reduction</u> of A to B

**Theorem:**

- <u>L</u>, <u>NL</u>, <u>P</u>, <u>NP</u>, <u>PSPACE</u> and <u>EXPTIME</u> are closed under log-space reductions.

# L Closed under $\leq_L$

WRONG!!

Why not simply apply $f$ then solve $A_2$ on the outcome?

**Claim:**

- $f$ is a LOGSPACE reduction from $A_1$ to $A_2$ and $A_2 \in L \Rightarrow A_1$ is in $L$

**Proof:**

- on input $x$: Simulate $M$ for $A_2$; whenever $M$ reads the $i^{th}$ symbol of its input, run $f$ on $x$ and wait for the $i^{th}$ bit to be outputted

# Graph Connectivity (CONN)

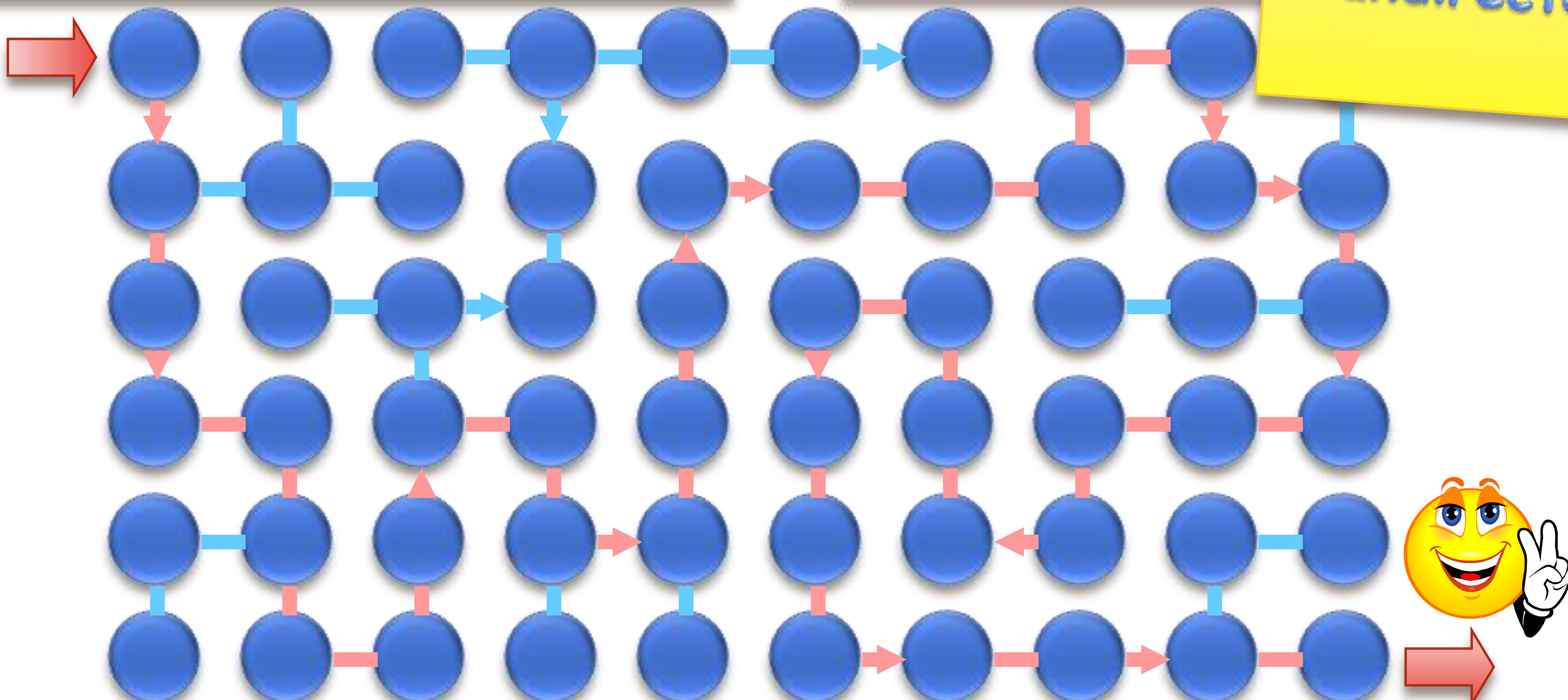
**Instance:**

- a directed graph $G=(V,E)$ and two vertices $s,t \in V$

**Decision Problem:**

- Is there a path from $s$ to $t$ in $G$?

- What about undirected?



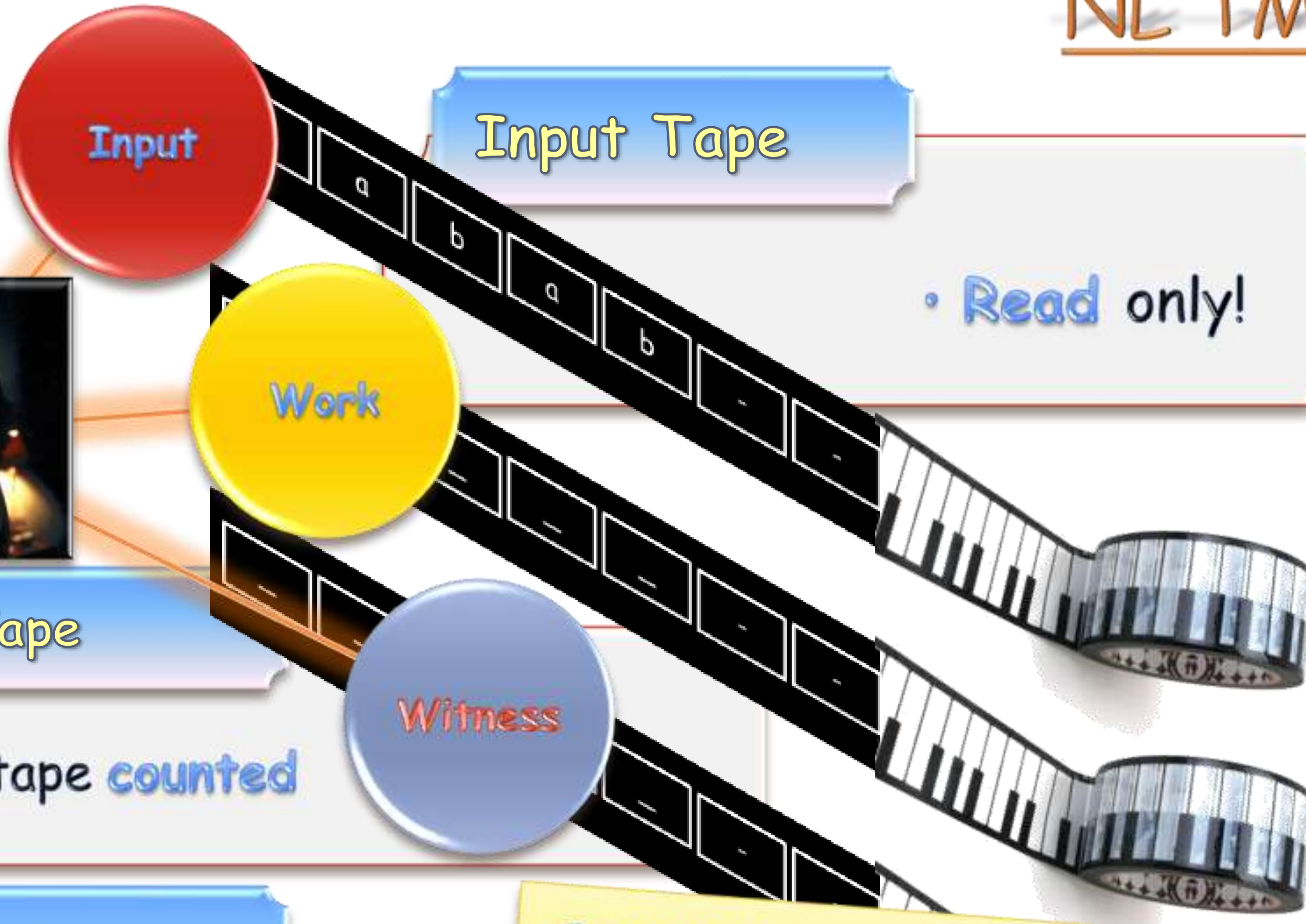
## CONN∈NL

1. Let u=s
2. Begin For i = 1, .., |V|
3. Let u= a (*non-deterministic*) neighbor of u
4. *accept* if u=t
5. End For
6. *reject* (did not reach t)

current position requires log|V| space

counting to |V| requires log|V| space

# NL TM




## Input Tape

- Read only!

## Work Tape

- Only tape counted

## Witness Tape

- Read only! No going back

**CONN witness:**
a path $s \mapsto t$

# CONN is NL-Complete

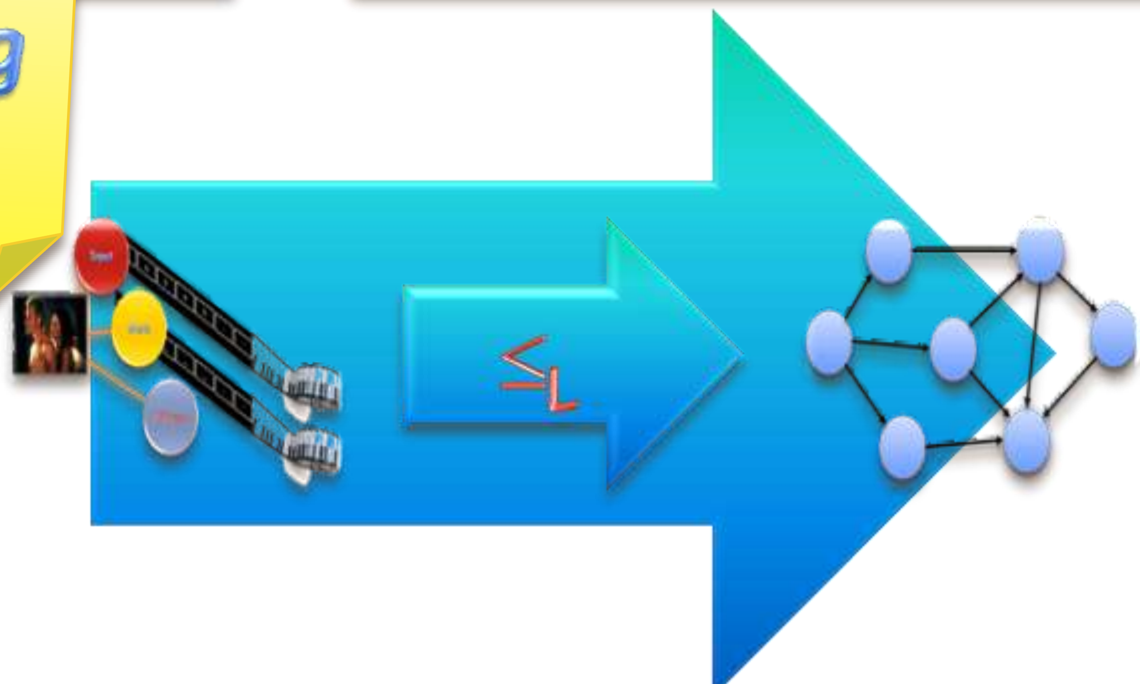
## Theorem:

- CONN is NL-hard

Assume a TM has 1 accepting configuration

## Proof:

- Given M,x, construct in LOGSPACE a CONN instance



# Define Configurations Graph: $G_{M,x}$

**$G_{M,x}$=(V, E)**

- For a (N)TM $M$ and input $x$

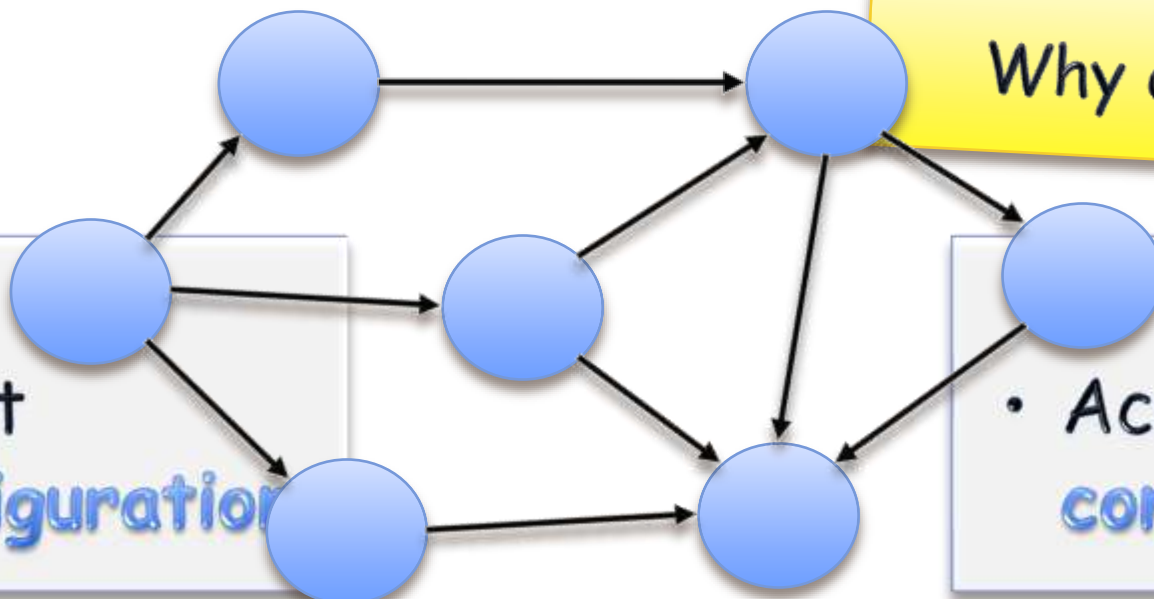
**V**

- All configurations

**E**

- $(u,v) \in E \Leftrightarrow \exists M,x$ transition $u \rightarrow v$

Why depend on $x$?



**s**

- Start configuration

**t**

- Accepting configuration

**Note:** $\forall M,x$: $M$ accepts $x$ $\Leftrightarrow$ $s \mapsto t$ in $G_{M,x}$

# CONN is NL-Complete

## Proof (end):

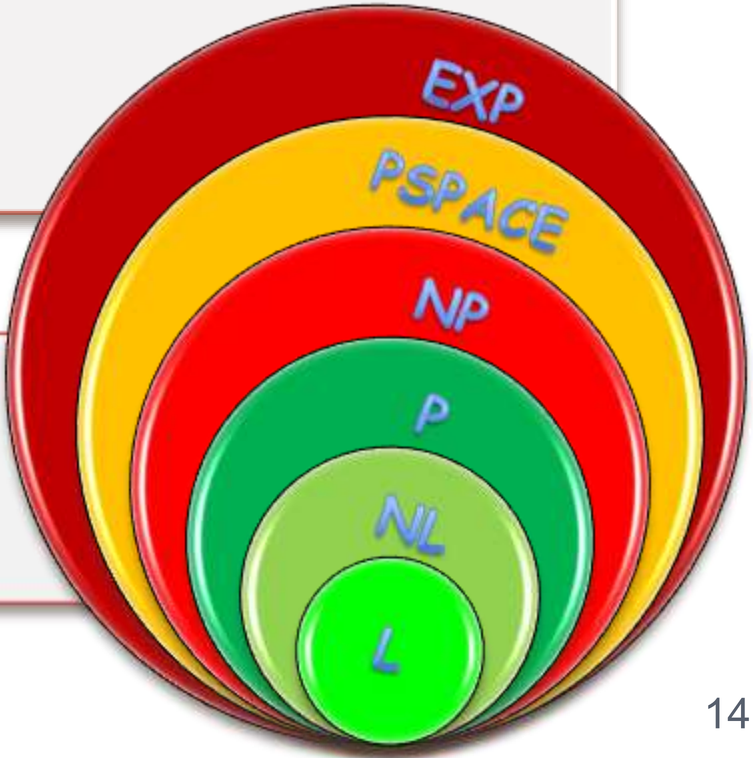
- $G_{M,x}$ can be constructed in Log-Space

## Corollary:

- NL⊆P

## Proof:

- CONN ∈P ■



**Now**

- that we have a language -CONN- representing NL

**We can**

- better analyze the complexity of space-bounded computations

# Savitch's Theorem

**Theorem:**

- $\forall S(n) \geq \log(n)$: $NSPACE[S(n)] \subseteq SPACE[S(n)^2]$

➡ NPSPACE=PSPACE

**Proof:**

- First $NL \subseteq SPACE[\log^2 n]$ then generalize

**Lemma:**

- $NL \subseteq DSPACE[\log^2 n]$

**Proof:**

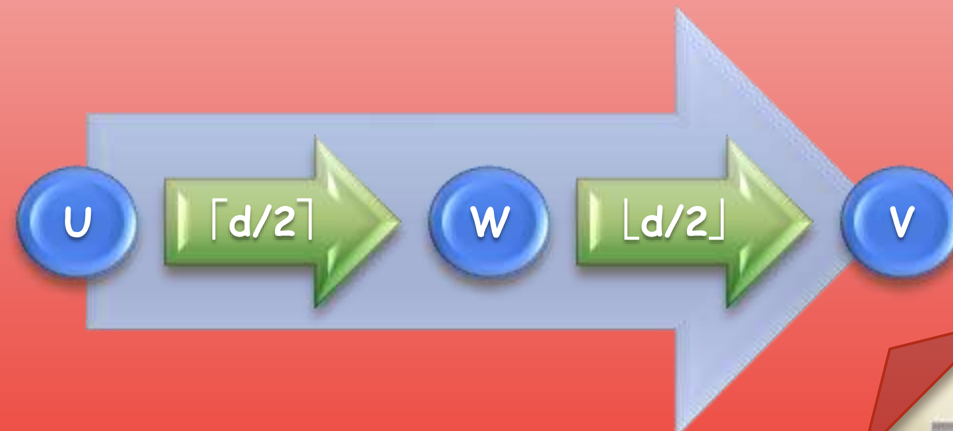
- Suffice to show $CONN \in DSPACE[\log^2 n]$

# CONN ∈ SPACE[$\log^2 n$]

$G=(V,E)$: is there a $d$-length path $u \mapsto v$?

```
Boolean PATH(u, v, d)
if (u, v) ∈ E return TRUE
if d=1 return FALSE
Begin For w ∈ V
   if PATH(u,w,⌈d/2⌉) and PATH(w,v,⌊d/2⌋) return TRUE
End For
return FALSE
```

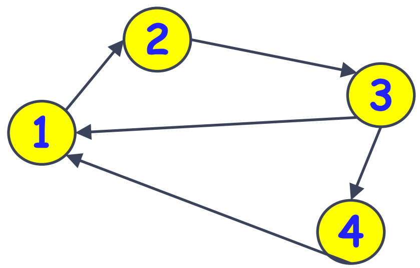
Is there a middle vertex $w$, s.t. $u \mapsto w$ and $w \mapsto v$, both of length $d/2$?




Recursion depth = $\log d$

$\log|V|$ space for each level

# Example of Savitch's algorithm




```
boolean PATH(a,b,d) {
    if there is an edge from a to b then
        return TRUE
    else {
        if (d=1) return FALSE
        for every vertex v (not a,b)  {
            if PATH(a,v, ⌈d/2⌉) and
               PATH(v,b, ⌊d/2⌋) then
               return TRUE
        }
        return FALSE
    }
}
```

(a,b,c)=Is there a path from a to b, that takes no more than c steps.

(1,4,3) TRUE

$3Log_2(d)$

# $O(\log^2 n)$-Space DTM for NL

**Proof (Lemma, end):**

- To solve CONN: call PATH(s,t,|V|)

**Have:**

- $NL \subseteq SPACE(\log^2 n)$

**Want:**

- $\forall S(n) \geq \log n$
  $NSPACE(S(n)) \subseteq SPACE(S^2(n))$

# Scale up

**Claim:**

- For any two space constructible functions $s_1(n)$, $s_2(n) \geq \log n$, $e(n) \geq n$:

**If**

- $NSPACE[s_1(n)] \subseteq SPACE[s_2(n)]$

**Then**

- $NSPACE[s_1(e(n))] \subseteq SPACE[s_2(e(n))]$

# Padding argument

**Claim:**

- For any two space constructible functions $s_1(n)$, $s_2(n) \geq \log n$, $e(n) \geq n$:

**If**

- $NSPACE[s_1(n)] \subseteq SPACE[s_2(n)]$

**Then**

- $NSPACE[s_1(e(n))] \subseteq SPACE[s_2(e(n))]$

## Proof:

- For $L \in NSPACE[s_1(e(n))]$, let $L^e = \{X\ \#^{e(|x|)-|x|} \mid X \in L\}$;

## Claim 1:

Premise

- $L^e \in NSPACE[s_1(n)] \subseteq DSPACE[s_2(n)]$

## Hence

- $\exists M'$ of $s_2(n)$-DSPACE for $L^e$

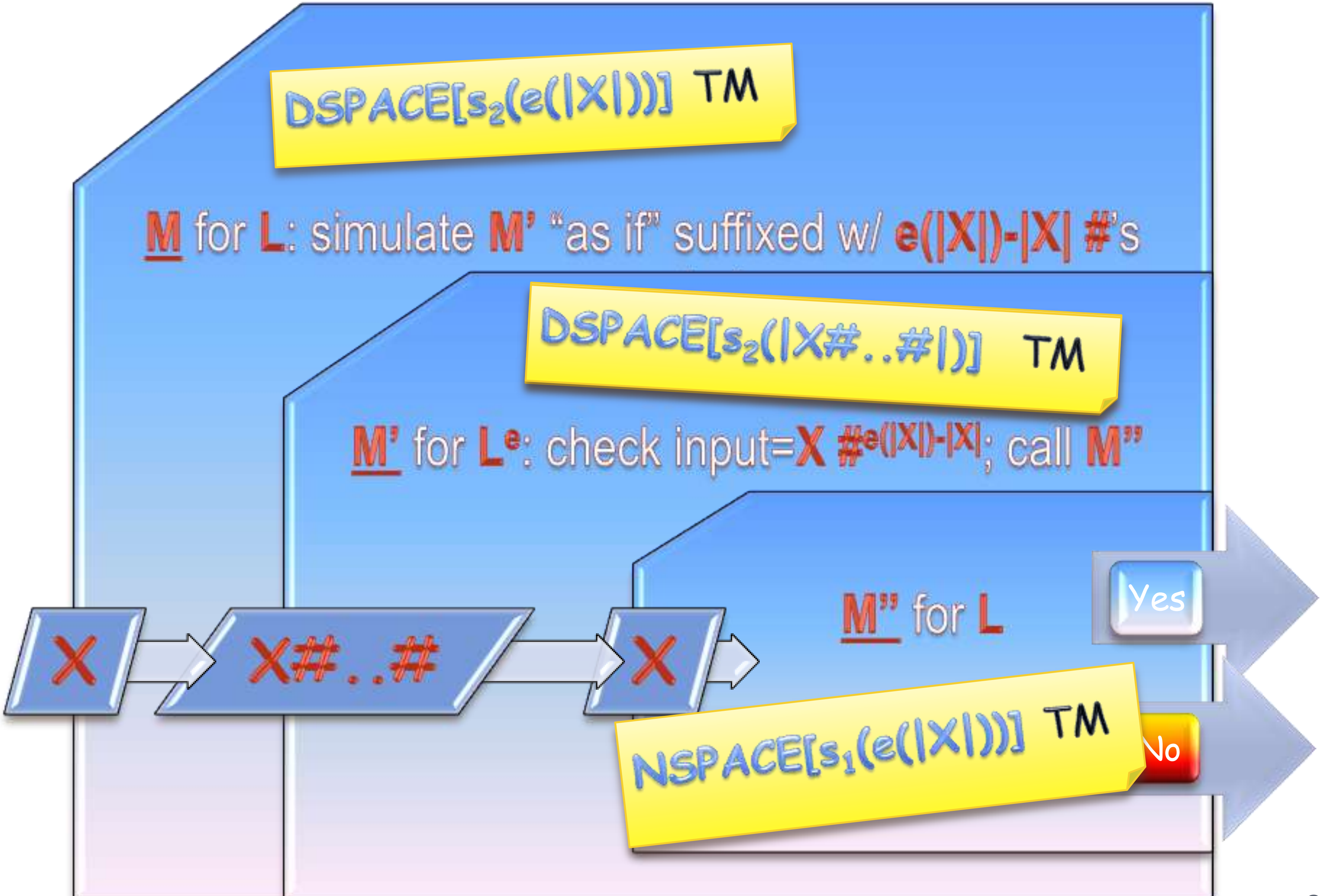
M' counts |X| and #'s to ensure proper form, then treat # as _

## Claim 2:

- $\exists M$ of $s_2(e(n))$-DSPACE for L

M simulates M' and "cheats" it to "see" e(|X|)-|X| extra #'s

# Padding



DSPACE[s2(e(|X|))] TM
M for L: simulate M' "as if" suffixed w/ e(|X|)-|X| #'s
DSPACE[s2(|X#..#|)] TM
M' for L^e: check input=X #^{e(|X|)-|X|}; call M"
M" for L
NSPACE[s1(e(|X|))] TM
Yes
No
X
X#..#
X

**So far**

- Simulation of **Non-deterministic space-bounded** computation does not incur very large overhead

**Next**

- What about **complementation**? **NL** vs. **coNL**

# NON-CONN

**NON-CONN Instance:**

- A directed graph G and two vertices s,t∈V

**Decision Problem:**

- Is there no path from s to t?

**Observation:**

- NON-CONN is coNL-Complete.

As CONN is NL-Complete

What if we prove non-CONN is in NL?

# Immerman/Szelepcsényi: coNL = NL

**Theorem:**

- Non-CONN ∈ NL

**Proof:**

**Def** reachable(G) ≡ { v | s↦v }

**Def** let $G_{-t}$ ≡ (V, E - V×{t} )

**Witness:** #reachable(G) = #reachable($G_{-t}$)

**Suffice:** #reachable(G) = r

**Def** $reachable_l$ ≡ { v | s↦v of length ≤l }

**Induction:** $r_l$=#$reachable_l$ Base: $r_0$=1

## Induction step:

- Extend an NL-verifiable witness W to "$r_l$=#reachable$_l$" to a witness to "$r_{l+1}$=#reachable$_{l+1}$":

W#$r_l$#

| | | |
|---|---|---|
| 1 | ∈/∉reachable$_{l+1}$ | \$$W_1$\$ |
| … | | |
| \|V\| | ∈/∉reachable$_{l+1}$ | \$$W_{\|V\|}$\$ |

$W_i$ for i∈reachable$_{l+1}$:

- s↦i of length ≤l+1

$W_i$ for i∉reachable$_{l+1}$:

| | | |
|---|---|---|
| 1 | ∈/∉reachable$_l$ | *$Z_1$* |
| … | | |
| \|V\| | ∈/∉reachable$_l$ | *$Z_{\|V\|}$* |

$Z_j$ for j∈reachable$_l$ only if j→i∉E:

- s↦j of length ≤l

$Z_j$ for j∉reachable$_l$: empty

# N.D. Algorithm for $reach_s(v, l)$

$reach_s(v, l)$

1. length = l; u = s

2. while (length > 0) {
   3. if u = v return 'YES'
   4. else, for all (u' ∈ V) {
      5. if (u, u') ∈E nondeterministic switch:
         5.1 u = u'; --length; break
         5.2 continue
   }
}
6. return 'NO'

Takes up logarithmic space

This N.D. algorithm might never stop

# N.D. Algorithm for $CR_S$

$CR_s(d)$

1. count = 0
2. for all $u \in V$ {
    3. $count_{d-1} = 0$
    4. for all $v \in V$ {
        5. nondeterministic switch:
            5.1 if reach(v, d - 1) then ++$count_{d-1}$ else *fail*
            if $(v,u) \in E$ then ++count; break
            5.2 continue
    }
    6. if $count_{d-1} < CR_s(d-1)$ fail
}
7. return count

Assume $(v,v) \in E$

Recursive call!

# N.D. Algorithm for CR

Main Algorithm:

$CR_s$

C = 1

for d = 1..|V|

C = CR(d, C)

return C

$CR_s$ ( d , C)

1. count = 0
2. for all $u \in V$ {
   3. $count_{d-1}$ = 0
   4. for all $v \in V$ {
      5. nondeterministic switch:
         5.1 if reach(v, d - 1) then ++$count_{d-1}$ else *fail*
         if $(v,u) \in E$ then ++count; break
         5.2 continue
   }
   6. if $count_{d-1}$ < C fail
}
7. return count

parameter

**Corollary**

- Space-bounded computation classes closed under complementation: $\forall s(n) \geq \log(n)$: $NSPACE(s(n)) = coNSPACE(s(n))$

padding argument

**Next**

- A basic problem complete for PSPACE

# TQBF

## Instance:

- a fully quantified Boolean formula $\phi$

## Decision Problem:

- Is $\phi$ true?

EG $\forall x \exists y \forall z[(x \vee \neg y \vee z) \wedge (\neg x \vee y)]$

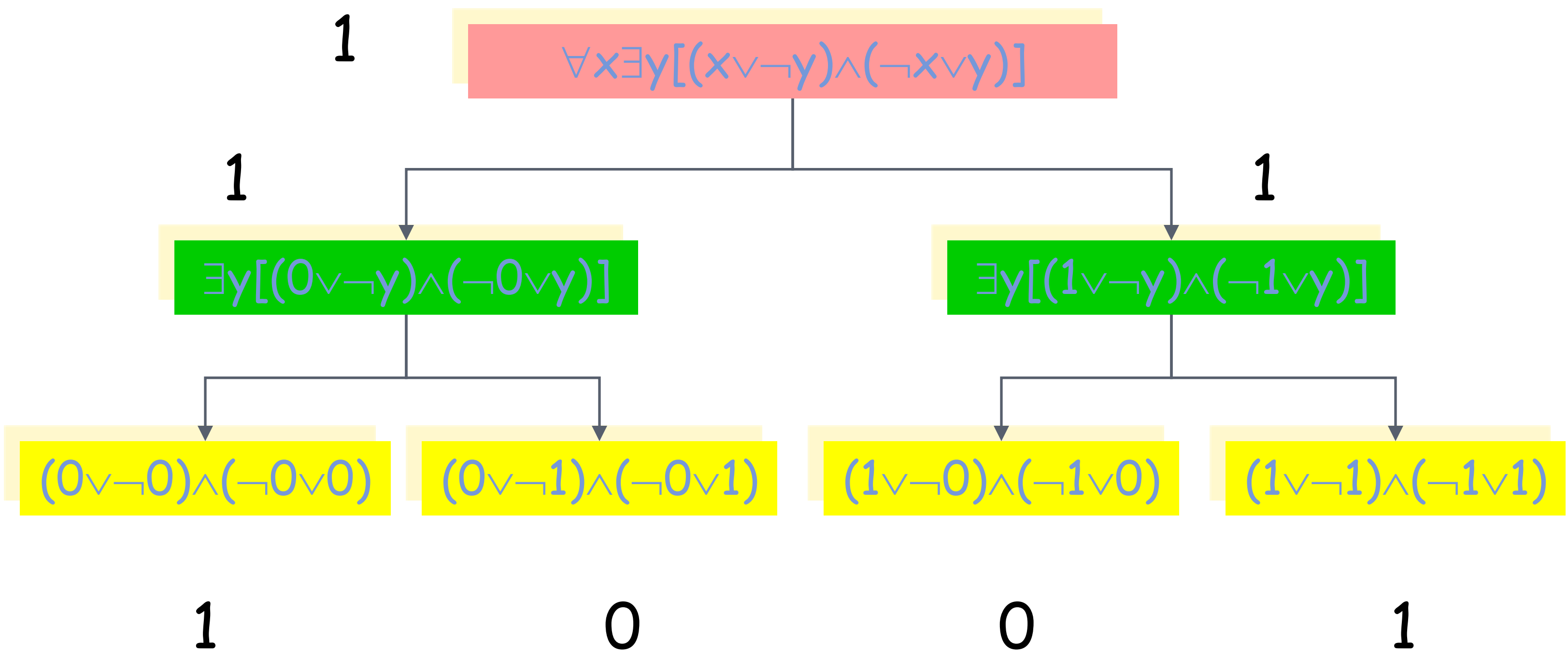
## Theorem:

- TQBF $\in$ PSPACE

## Proof:

- A poly-space algorithm $A$ that evaluates $\phi$:
  if $\phi$ is quantifier-free return its value
  if $\phi = \forall x. \psi(x, \ldots)$ return $A(\psi(0, \ldots)) \wedge A(\psi(1, \ldots))$
  if $\phi = \exists x. \psi(x, \ldots)$ return $A(\psi(0, \ldots)) \vee A(\psi(1, \ldots))$

# Algorithm for TQBF



# TQBF is PSPACE-Complete

$\in L_M$ ⇔ $\forall x_1 \exists x_2 \forall x_3 \ldots [\ldots]$

**Theorem:**

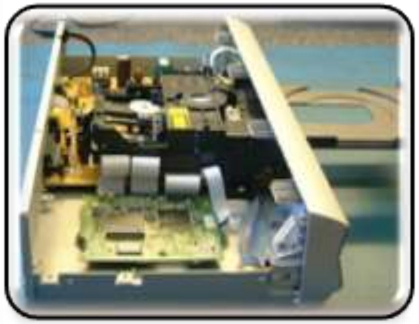
- TQBF is PSAPCE-hard

**Proof:**

- For a TM $M$, start with a BF
  $transition_M(\underline{u}, \underline{v})$ ⇔ on $\underline{u}$ $M$ moves to $\underline{v}$
  Construct, inductively, the BF
  $\phi_M(\underline{u},\underline{v},d)$ ⇔ on $\underline{u}$, $M$ arrives at $\underline{v}$ in $\leq 2^d$ steps:
  $\phi_M(\underline{u},\underline{v},0) \equiv transition_M(\underline{u}, \underline{v}) \vee \underline{u}=\underline{v}$
  $\phi_M(\underline{u},\underline{v},d) \equiv \exists \underline{w} \forall \underline{u}' \forall \underline{v}'$
  $[\ ((\underline{u}'=\underline{u} \wedge \underline{v}'=\underline{w}) \vee (\underline{u}'=\underline{w} \wedge \underline{v}'=v)) \Rightarrow \phi_M(\underline{u}',\underline{v}',d-1)\ ]$
- $f(M, x) = \phi_M(start[x], accept, \lceil lg\#of\ config. \rceil)$

$\underline{u}$, $\underline{v}$ vectors describing configurations

$transition_M(\underline{u}, \underline{v})$ almost equality



Defined space-complexity, in particular, the complexity classes: **L**, **NL**, **coNP**, **PSPACE**.

Proved:


**Completeness**:

**CONN** for **NL**; **TQBF** for **PSPACE**

Savitch's theorem (**NL⊆SPACE(log²)**)

The **padding argument** (scaling up)

Immerman's theorem (**NL=coNL**)

WWindex

Space Complexity

Savitch's Theorem

Log Space Reductions

Immerman's Theorem

TQBF

Complexity

L

NL

PSPACE

Savitch, Walter

Immerman, Neil

Róbert Szelepcsényi