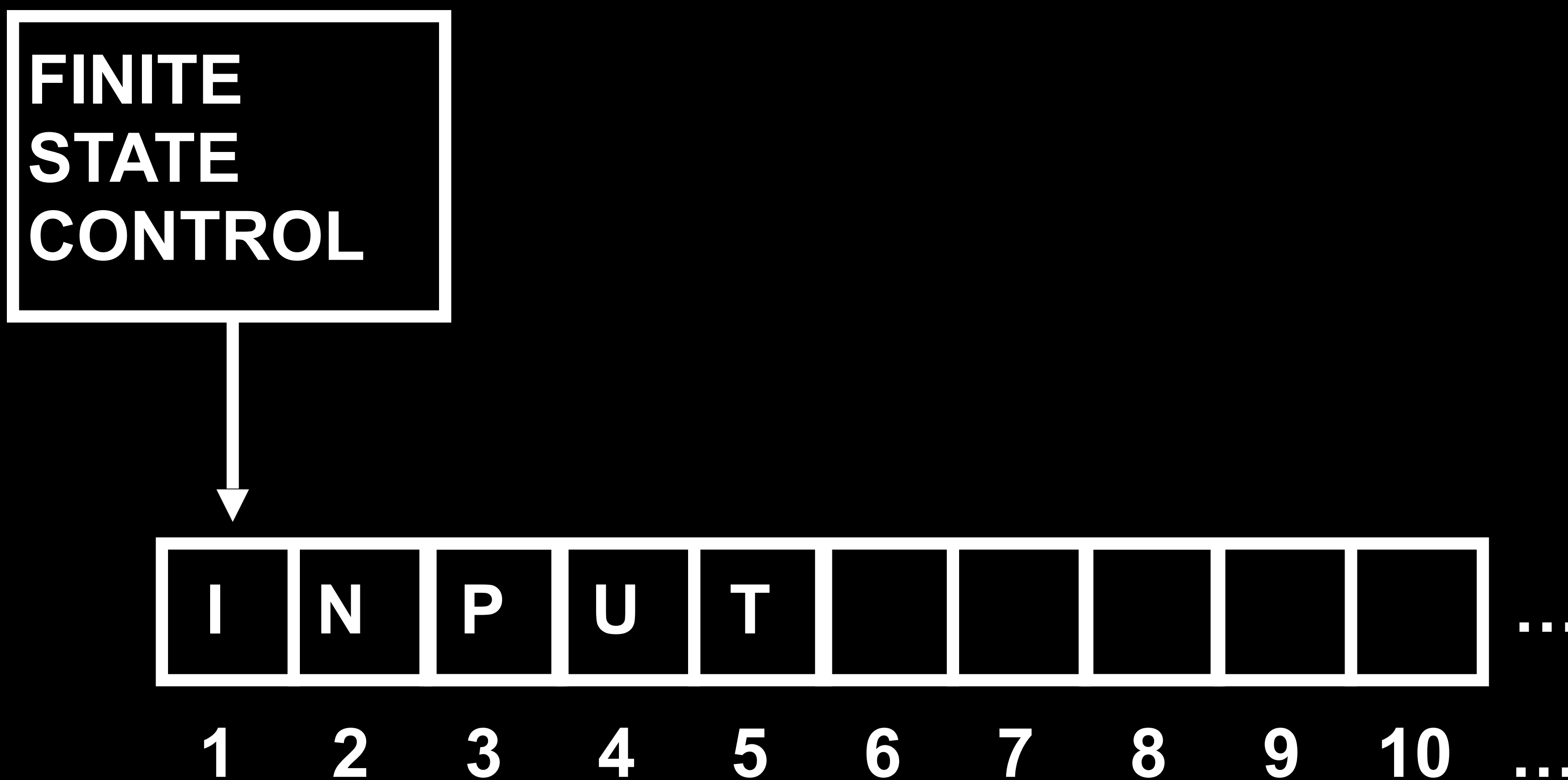
Space Complexity: Savitch's Theorem and PSPACE-Completeness

Tuesday April 15

MEASURING SPACE COMPLEXITY



We measure space complexity by looking at the furthest tape cell reached during the computation

Let M = deterministic TM that halts on all inputs.

Definition: The **space complexity** of M is the function s : $\mathbb{N} \rightarrow \mathbb{N}$, where s(n) is the furthest tape cell reached by M on any input of length n.

Let N be a **non-deterministic** TM that halts on all inputs in all of its possible branches.

Definition: The **space complexity** of N is the function s : $\mathbb{N} \rightarrow \mathbb{N}$, where s(n) is the furthest tape cell reached by M, on **any branch if its computation**, on any input of length n.

Definition: SPACE(s(n)) =
{ L | L is a language decided by a O(s(n)) space deterministic Turing Machine }

Definition: NSPACE(t(n)) =
{ L | L is a language decided by a O(s(n)) space **non-deterministic** Turing Machine }

$$\mathbf{PSPACE} = \bigcup_{k \in \mathbf{N}} \mathbf{SPACE}(n^k)$$

$$\mathbf{NPSPACE} = \bigcup_{k \in \mathbf{N}} \mathbf{NSPACE}(n^k)$$

3SAT ∈ **SPACE(n)** ⊂ **PSPACE**

(	x	∨	¬	y	∨	x	)		(	y	∨	x	∨	y	)						
(	x	∨	¬	y	∨	x	)		(	y	∨	x	∨	y	)		#	x		y	
(	x	∨	¬	y	∨	x	)		(	y	∨	x	∨	y	)		#	x	0	y	0
(	x	∨	¬	y	∨	x	)		(	y	∨	x	∨	y	)		#	x	0	y	1
(	x	∨	¬	y	∨	x	)		(	y	∨	x	∨	y	)		#	x	1	y	0

Assume a deterministic Turing machine that halts on all inputs runs in space s(n)

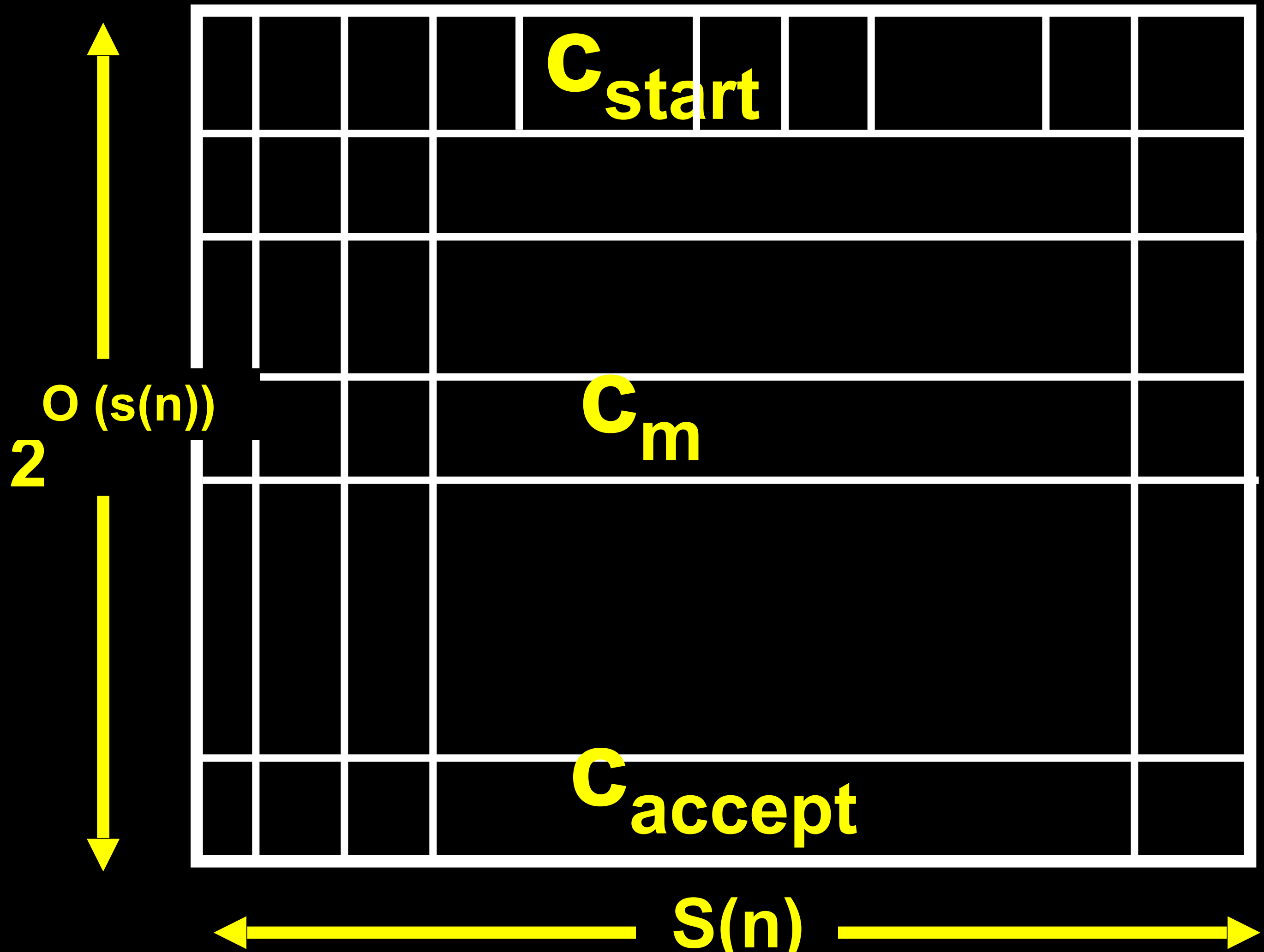
Question: What's an upper bound on the number of time steps for this machine?

A configuration gives a head position, state, and tape contents. Number of configurations is at most:

$$s(n)\ |Q|\ |\Gamma|^{s(n)} = 2^{O(s(n))}$$

Number of Configurations

$$s(n)\,|Q|\,|\Gamma|^{s(n)} = 2^{O(s(n))}$$



MORAL:
Space S computations can be simulated in at most $2^{O(S)}$ time steps

PSPACE $\subseteq$ EXPTIME

$$\text{EXPTIME} = \bigcup_{k \in N} \text{TIME}(2^{n^k})$$

MORAL:

Space S computations can be simulated in at most $2^{O(S)}$ time steps

$$L \subset NL \subset P$$

Any function computable in log space is also in polynomial time.

- S-T-Connectivity (STCONN):

- S-T-Connectivity (STCONN): given directed graph G = (V, E) and nodes s, t, is there a path from s to t ?

STCONN is in NL

- NUMSTEPS = 0 (number of steps taken.)
- C = s (current node)
- FLAG=False

- Until NUMSTES = n do
  - GUESS Z from 1 to n
  - Increment NUMSTEPS
  - If (c,z) is an edge in G, set c=z
  - If c==t set FLAG= True.

I started at S

– I got drunk

– and now I am at T

I wandered,

therefore, my path from S to T exists.

- **NSPACE(f(n))** = languages decidable by a multi-tape NTM that touches at most f(n) squares of its work tapes *along any computation path*, where n is the input length, and f :**N** ! **N**

Let C configuration graph for a space f(n) NTM on input x.

C has $c^{f(n)} = 2^{kf(n)}$ nodes (Exponential in f(n))

f(n) = k'log(n) means POLY-SIZED graph.

STCONN is NL-Hard under logspace reductions

- Proof:
  - given $L \in$ **NL** decided by NTM M construct configuration graph for M on input x (can be done in logspace),
  - s = starting configuration; t = q_{accept}
  - Output graph as a list of edges.

Savitch's Theorem

<u>Theorem</u>: STCONN $\in$ **SPACE($\log^2$ n)**

- Corollary: **NL $\subset$ SPACE($\log^2$n)**

Proof of Theorem

- input: G = (V, E), two nodes s and t
- recursive algorithm:

```
/* return true iff path from x to y of length at most 2^i */

PATH(x, y, i)
 if i = 0 return ( x = y or (x, y) ∈ E )          /* base case */
 for z in V
    if PATH(x, z, i-1) and PATH(z, y, i-1) return(true);
 return(false);
end
```

Proof of Theorem

- answer to STCONN: PATH(s, t, log n)
- space used:
  - (depth of recursion) x (size of “stack record”)
- depth = log n
- claim stack record: “(x, y, i)” sufficient
  - size O(log n)
- when return from PATH(a, b, i) can figure out what to do next from record (a, b, i) and previous record

Savitch's Theorem

<u>Theorem</u>: STCONN $\in$ **SPACE($\log^2 n$)**

- Corollary:
- **NSPACE(s(n)) $\subseteq$ SPACE(s(n)^2)**

s(n) $\geq$ log n

- **NSPACE(f(n))** = languages decidable by a multi-tape NTM that touches at most f(n) squares of its work tapes *along any computation path*, where n is the input length, and f :**N** ! **N**

Let C configuration graph for a space f(n) NTM M on input x.

C has $c^{f(n)}$ = $2^{kf(n)}$ nodes (Exponential in f(n))

M accepts x iff Start and Accept are connected in the directed graph C

USe Savirch's algorithm on C

space(log^2 ($2^{kf(n)}$)) = SPACE(f(n)^2)

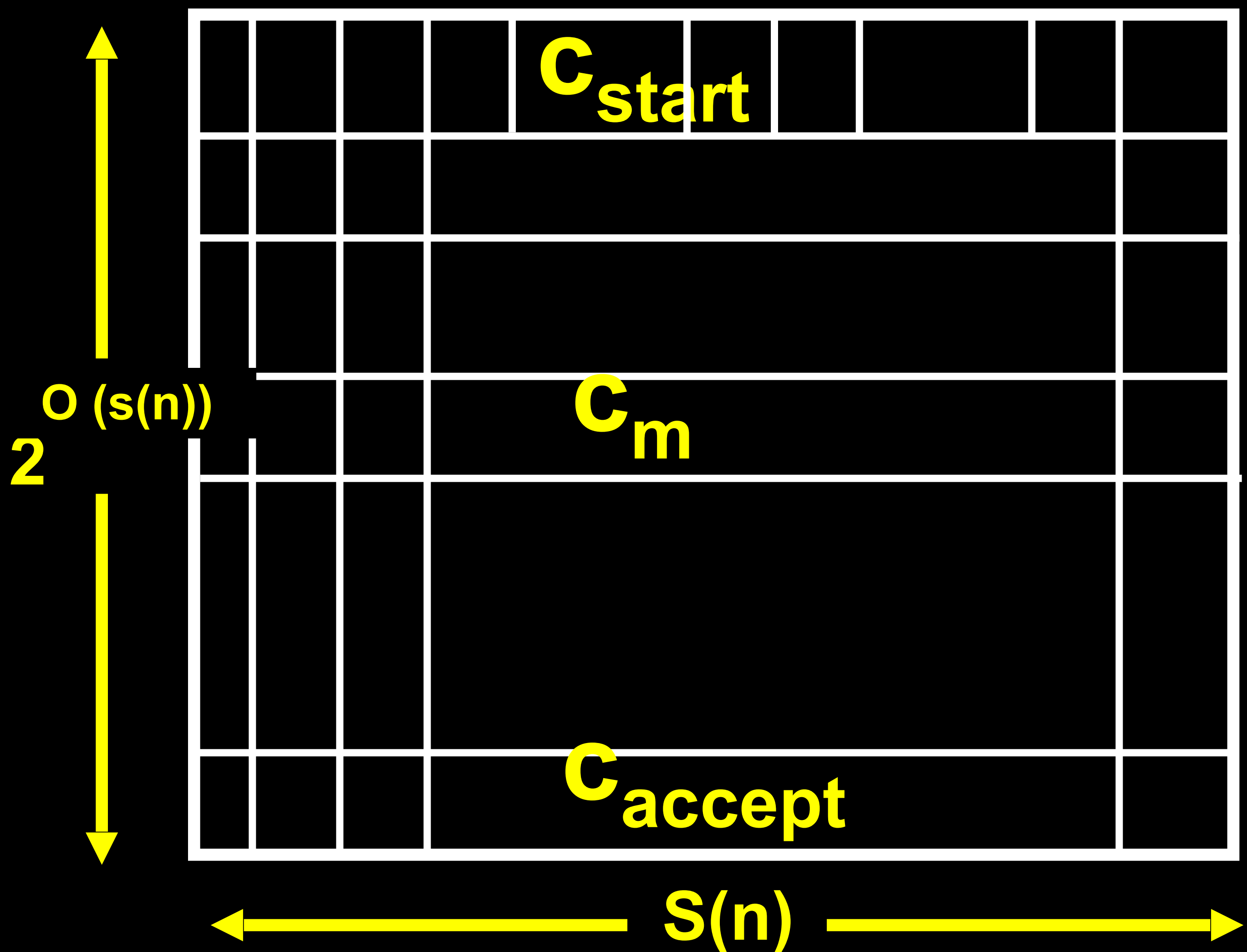
Savitch's Theorem

<u>Theorem</u>: STCONN $\in$ **SPACE($\log^2 n$)**

- Corollary: **NL $\subset$ SPACE($\log^2 n$)**
- Corollary: **NPSPACE = PSPACE**

Number of Configurations

$$s(n)\ |Q|\ |\Gamma|^{s(n)} = 2^{O(s(n))}$$



Theorem: For a function s where $s(n) \geq n$

$$NSPACE(s(n)) \subseteq SPACE(s(n)^2)$$

Proof:

Let N be a nondeterministic TM using s(n) space

Modify N so that when it accepts, it goes to a special state q_s, clears its tape, and moves its head to the leftmost cell

N has a UNIQUE accepting configuration: $C_{acc} = q_s \sqcup \ldots \sqcup$

Construct a deterministic M that on input w, runs CANYIELD(C_0, C_{acc}, $2^{ds(|w|)}$)

Here d > 0 is chosen so that $2^{d\,s(|w|)}$ upper bounds the number of configurations of N(w)

=> $2^{ds(|w|)}$ is an upper bound on the running time of N(w).

Theorem: For a function s where $s(n) \geq n$

$$\text{NSPACE}(s(n)) \subseteq \text{SPACE}(s(n)^2)$$

Proof:

Let N be a nondeterministic TM using s(n) space

Modify N so that when it accepts, it goes to a special state q_s, clears its tape, and moves its head to the leftmost cell

N has a UNIQUE accepting configuration: $C_{acc} = q_s \sqcup \ldots \sqcup$

Construct a deterministic M that on input w, runs CANYIELD(C_0, C_{acc}, $2^{ds(|w|)}$)

Why does it take only $s(n)^2$ space?

Theorem: For a function s where $s(n) \geq n$

$$NSPACE(s(n)) \subseteq SPACE(s(n)^2)$$

Proof:

Let N be a nondeterministic TM using $s(n)$ space

Modify N so that when it accepts, it goes to a special state q_s, clears its tape, and moves its head to the leftmost cell

N has a UNIQUE accepting configuration: $C_{acc} = q_s \square \ldots \square$

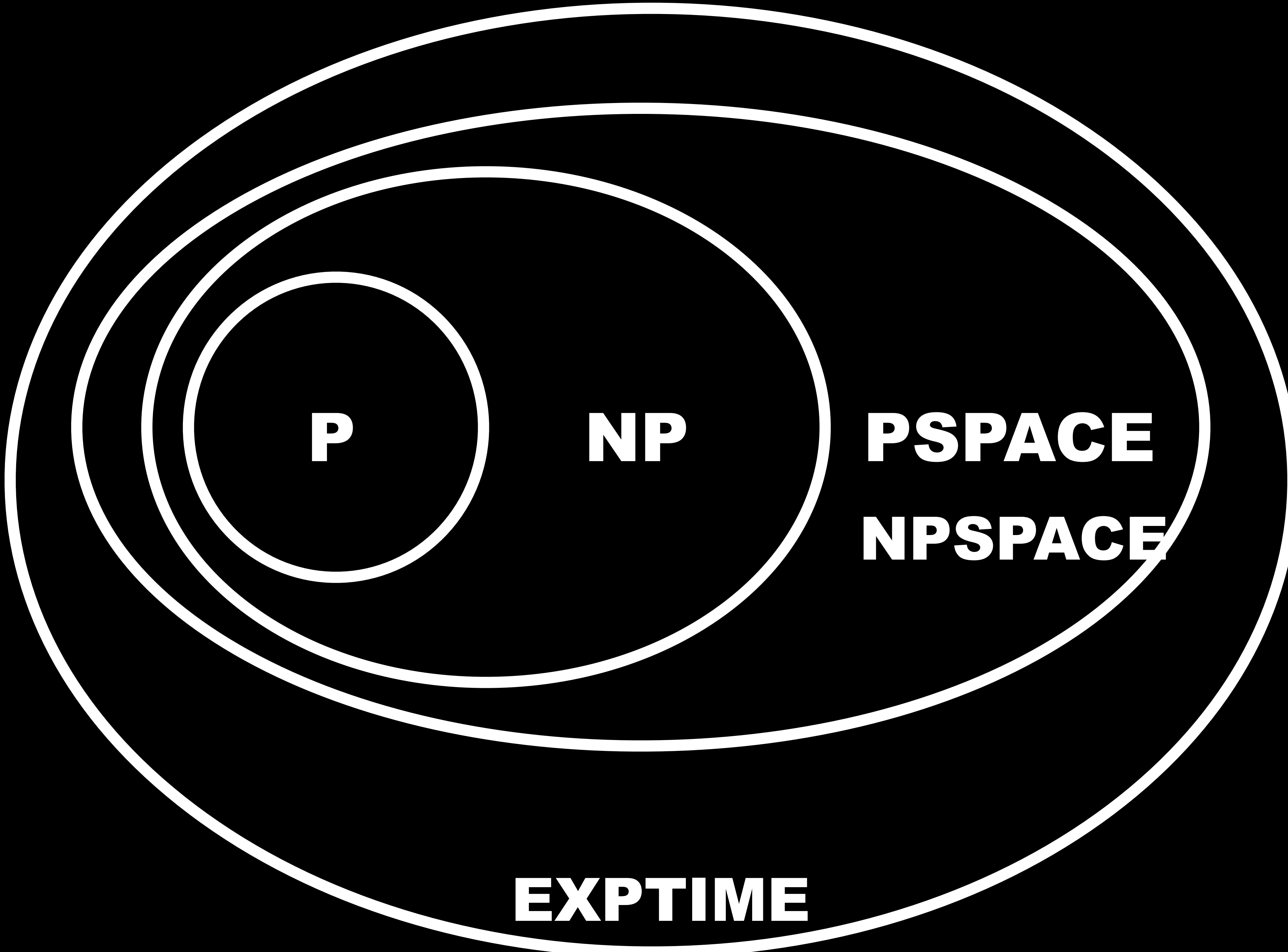
Construct a deterministic M that on input w,
runs CANYIELD(C_0, C_{acc}, $2^{ds(|w|)}$)

Uses $\log(2^{d\,s(|w|)})$ recursions. Each level of recursion uses $O(s(n))$ extra space. Therefore uses $O(s(n)^2)$ space!

$$\mathrm{PSPACE} = \bigcup_{k \in \mathbb{N}} \mathrm{SPACE}(n^k)$$

$$\mathrm{NPSPACE} = \bigcup_{k \in \mathbb{N}} \mathrm{NSPACE}(n^k)$$

PSPACE = NPSPACE

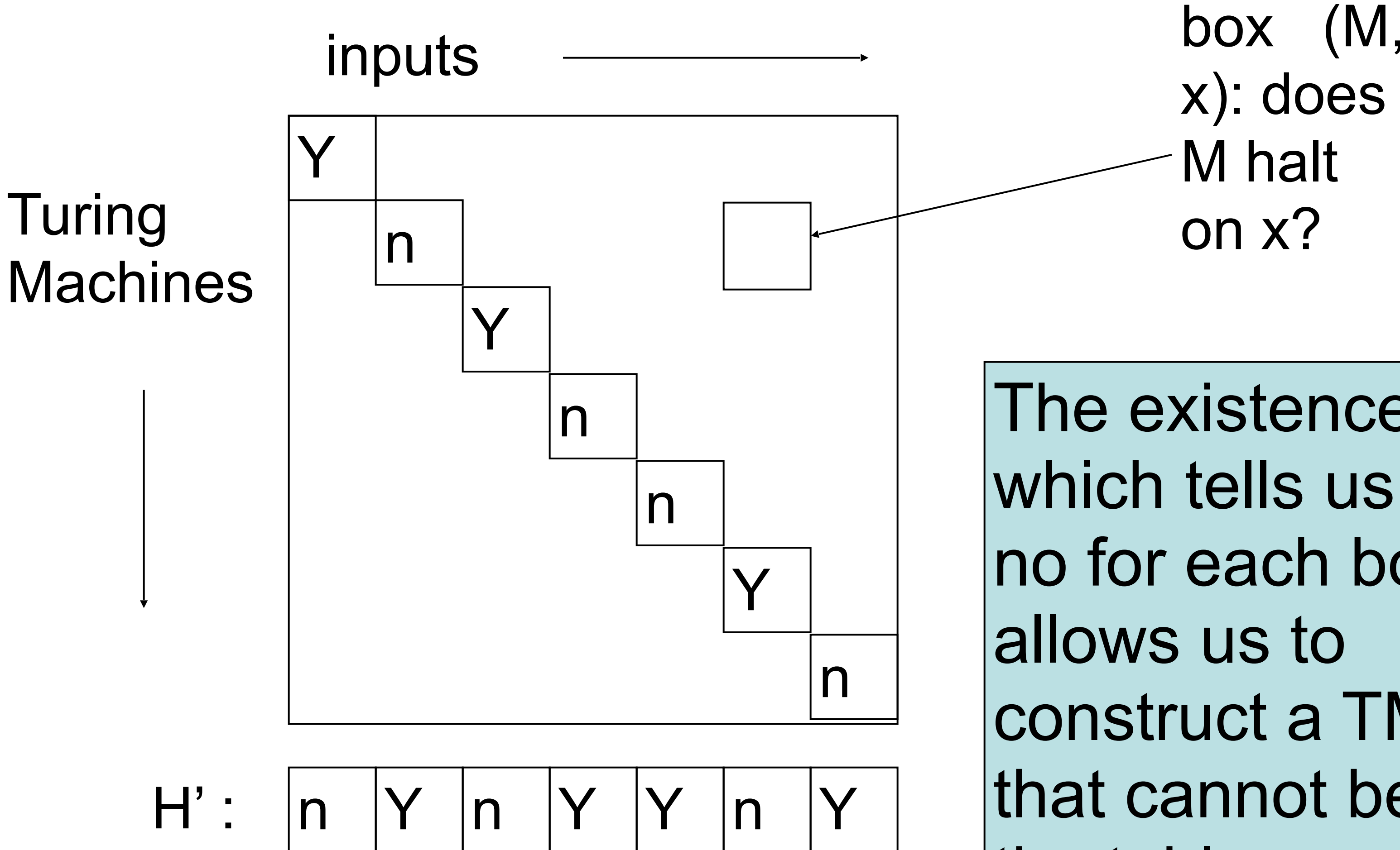

P
NP
PSPACE
NPSPACE
EXPTIME

P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ EXPTIME

P ≠ EXPTIME

TIME HIERARCHY THEOREM

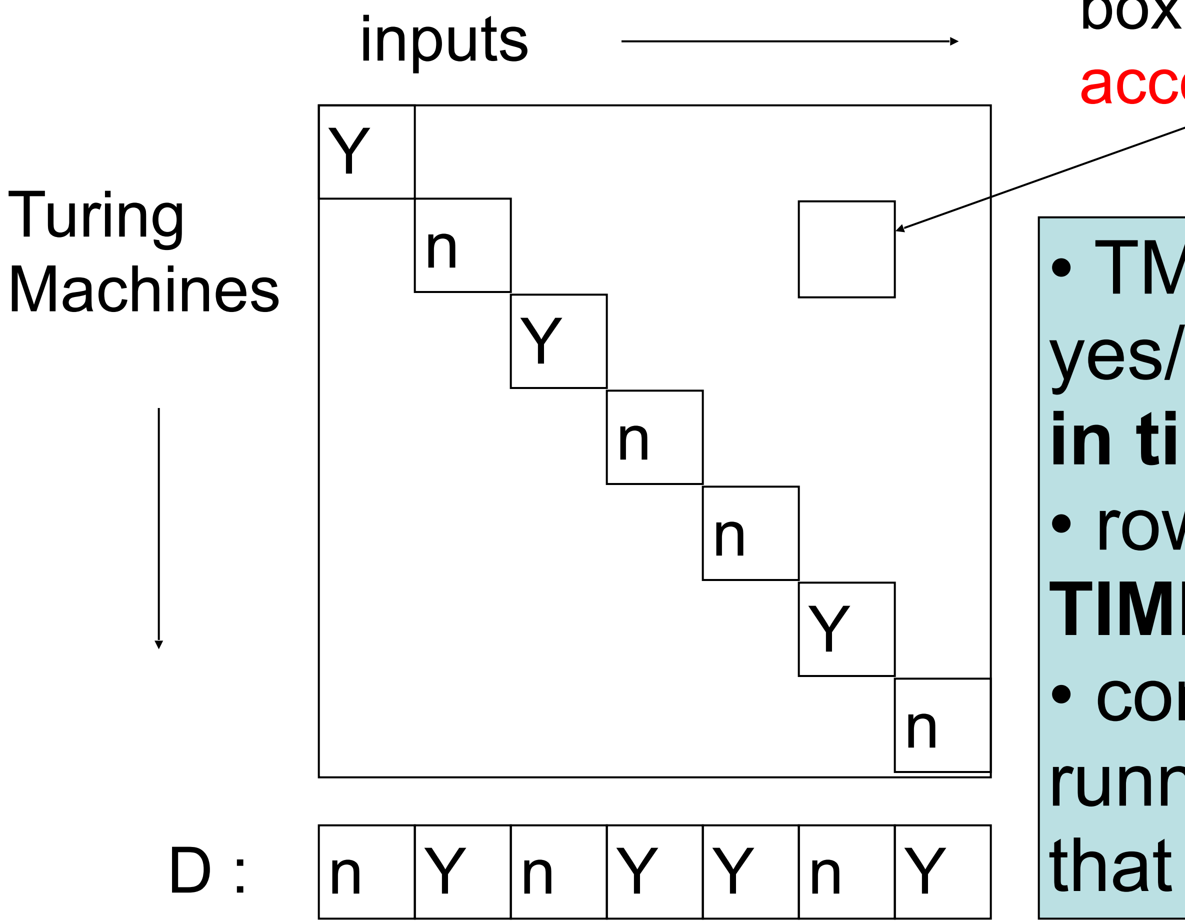
Recall proof for Halting Problem



The existence of H which tells us yes/no for each box allows us to construct a TM H' that cannot be in the table.

Time Hierarchy Theorem

box (M, x): does M
accept x in time f(n)?



- TM SIM tells us yes/no for each box **in time g(n)**
- rows include all of **TIME(f(n))**
- construct TM D running in time g(2n) that is not in table

Time Hierarchy Theorem

<u>Theorem</u> (Time Hierarchy Theorem): For every *proper complexity function* $f(n) \geq n$:

$$\mathbf{TIME(f(n))\ (\ TIME(f(2n)^3).}$$

Proof of Time Hierarchy Theorem

- Claim: there is a TM SIM that decides

  {<M, x> : M accepts x in ≤ f(|x|) steps}

  and runs in time $g(n) = f(n)^3$.
- Proof sketch: SIM has 4 work tapes
  - contents and "virtual head" positions for M's tapes
  - M's transition function and state
  - f(|x|) "+"s used as a clock
  - scratch space

Proof of Time Hierarchy Theorem

    - contents and “virtual head” positions for M’s tapes
    - M’s transition function and state
    - f(|x|) “+”s used as a clock
    - scratch space
  - initialize tapes
  - simulate step of M, advance head on tape 3; repeat.
  - can check running time is as claimed.
- Important detail: need to initialize tape 3 in time O(f(n))

Proof of Time Hierarchy Theorem

- Proof:
  - SIM is TM deciding language

    { <M, x> : M accepts x in $\leq f(|x|)$ steps }
  - Claim: SIM runs in time $g(n) = f(n)^3$.
  - define new TM D: on input <M>
    - if SIM accepts <M, M>, reject
    - if SIM rejects <M, M>, accept
  - D runs in time g(2n)

Proof of Time Hierarchy Theorem

- Proof (continued):
  - suppose M in **TIME(f(n))** decides L(D)
    - M(<M>) = SIM(<M, M>) ≠ D(<M>)
    - but M(<M>) = D(<M>)
  - contradiction.

Proper Complexity Functions

- Definition: f is a **proper complexity function** if
  - $f(n) \geq f(n-1)$ for all n
  - there exists a TM M that outputs exactly f(n) symbols on input 1^n, and runs in time $O(f(n) + n)$ and space $O(f(n))$.

Proper Complexity Functions

- includes all reasonable functions we will work with
  - $\log n$, $\sqrt{n}$, n^2, 2^n, $n!$, …
  - if f and g are proper then $f + g$, fg, $f(g)$, f^g, 2^g are all proper
- can mostly ignore, but be aware it is a genuine concern:
- Theorem: 9 non-proper f such that **TIME(f(n)) = TIME($2^{f(n)}$)**.

Best Hierarchy Theorems

<u>Theorem</u> (Time Hierarchy Theorem): For every *proper complexity function* f(n) ≥ n:

TIME(f(n)) (TIME(ω (f(n)log(f(n))).

<u>Theorem</u> (Space Hierarchy Theorem): For every *proper complexity function* f(n) ≥ log n:

SPACE(f(n)) (SPACE(ω f(n)).

ω

Time and Space Classes

L = SPACE(log n)

PSPACE = $\cup_k$ SPACE(n^k)

P = $\cup_k$ TIME(n^k)

EXP = $\cup_k$ TIME(2^{n^k})

I started at S

– I got drunk ← NL lucky, tiny brain

– and now I am NOT at T

How might I know that

NO PATH from S to T exists ???? !