Last time: relations between different resources.

For f a p.c.f., have:

- $TIME(f(n)) \subseteq SPACE(f(n))$
- $TIME(f(n)) \subseteq NTIME(f(n))$
- $SPACE(f(n)) \subseteq NSPACE(f(n))$
- $NTIME(f(n)) \subseteq \bigcup_{k>1} TIME(k^{f(n)})$
- $NTIME(f(n)) \subseteq SPACE(f(n))$
- $NSPACE(f(n)) \subseteq \bigcup_{k>1} TIME(k^{f(n)+\log n})$.

AB Chap. 4, Sipser Chap. 8, Cai Chap. 3

Today: start space complexity unit

- nondet space, Savitch's Theorem
  f a p.c.f.: $NSPACE(f) \subseteq SPACE(f^2)$
- NL, NL-completeness

Questions?

Recall basics abt space:

- $L \in SPACE(f(n))$ if some (multitape) TM

always halts

decides L & runs in space $\leq f(n)$ $\forall$ input $|x|=n$.

- Lin. compression for space: if $L \in SPACE(f(n))$, then also $\forall \varepsilon > 0$ $L \in SPACE(\varepsilon f(n))$ $\max\{1, \varepsilon f(n)\}$

- For TMs that produce output (compute a fn), we don't "count" output tape used towards TM's space cxity.

"Interesting" space classes: $\log(n)$ & up;
with $o(\log n)$ space, can't
- count to n
- write down loc. of input head in $|x|=n$
- " " " a single graph vertex's name,

} etc.

With $O(\log n)$ space, can maintain $O(1)$ many
counters
pointers/fingers into G
etc.

Key space classes:
- $L = SPACE(\log n)$
- $NL = NSPACE(\log n)$
- $PSPACE = \bigcup_{k>1} SPACE(n^k)$
- $NPSPACE = \bigcup_{k>1} NSPACE(n^k)$

PSPACE

SPACE CAN BE REUSED

Nondet Space

don't need this restric., actually

Thm (Savitch): For any p.c.f. $f(n) \geq \log n$,

$$NSPACE(f(n)) \subseteq SPACE(f(n)^2).$$

REACH : the key.

We know: REACH $\in$ P (BFS, DFS, etc)
by "marking reachable nodes" "
↳ such algs use linear space "

Stronger result: (we showed last time NL $\subseteq$ P)

Claim: REACH $\in$ NL.

Pf: Guess path (length $\leq n$), check it's a valid $s \rightsquigarrow t$ path.
do it one edge at a time

3-tape TM:
- tape 1: current node i — $\log n$ space
- tape 2: next node j (TM nondet. writes j) — "
- tape 3: counter up to n — "

TM (repeatedly)
- writes (nondet) j on 2nd tape
- reads input to confirm $i \to j$ edge present in G

- updates counter
- replaces i with j.

Halts & rejects when counter hits n
accepts if $i = t$ (end node $s \rightsquigarrow t$ path). ■

In fact...

Claim: $REACH \in SPACE(\log^2 n)$.

Idea: recursion + $\exists$ midpoints.

Pf: Let $G = (V, E)$ n-node digraph. (G, s, t)

Define

$$PATH(x, y, i) = \begin{cases} \text{TRUE if } G \text{ cont. dir. } x \rightsquigarrow y \text{ path of length} \le 2^i \\ \text{FALSE otherwise.} \end{cases}$$

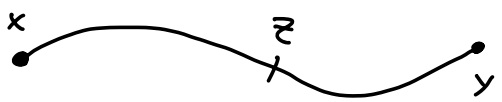
So $G \in REACH$ iff $PATH(s, t, \log n)$

Goal: decide $PATH(x, y, i)$ using small space.

Idea: $\exists$ $x \overset{2^i}{\rightsquigarrow} y$ iff there exists some z (midpoint) s.t.

$$x \overset{2^{i-1}}{\rightsquigarrow} z, \quad z \overset{2^{i-1}}{\rightsquigarrow} y.$$

Recurse, space-efficiently.



Here's a rec. alg. for PATH:

PATH (x, y, i)

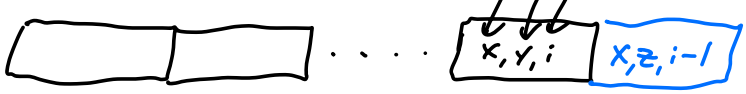
- if $i = 0$ then if $x \to y$ edge is in G, return TRUE
  else return FALSE;
- else $(i > 0)$
  for each vtx $z \in V$
  - if PATH$(x, z, i-1)$ & PATH$(z, y, i-1)$ return TRUE

  (there was no such z)
  return FALSE

This is a correct alg. Can be run in $\log^2 n$ space on a TM as follows:

- tape 2: $O(1)$ counters, scratch space, etc; $O(\log n)$ space.
- tape 1: "stack" of recursive history of calls.

Contains various 3-tuples, each is $3 \log n$ bits



Reuse space as you go through & check various z's.
Once PATH$(x, z, i-1)$ has been computed: • if TRUE, erase $x, z, i-1$ & replace with $z, y, i-1$ & keep going:

if PATH$(z, y, i-1)$ found to be T, erase both $z, y, i-1$ & x, y, i & now know PATH(x, y, i) is T ...

- if $PATH(x,z,i-1)$ was FALSE, erase $x,z,i-1$ & replace on tape with $x,z',i-1$ for next $z' \in V$. If all z' used: erase x,y,i (b/c found $PATH(x,y,i)$ is FALSE).

Use i levels of recursion on $PATH(x,y,i)$, $O(\log n)$ bits of space per level of recursion.

So total space usage $= O(i \cdot \log n)$ for $PATH(x,y,i)$ on (n-node G)

So $PATH(s,t,\log n)$ computed by above alg in $O(\log^2 n)$ space. ∎

Runtime? $T(i) =$ time for $PATH(x,y,i)$ on (n-node G).

$T(i) = 2n \cdot T(i-1)$ so $T(i) = (2n)^i$

& $T(\log n) = (2n)^{\log n} = 2^{\log n} \cdot n^{\log n} = n^{\log(n)+1}$.

Q: is there a $\text{poly}(n)$ time, $\log^2 n$ space alg for REACH? Unknown....

Now pf of

Thm (Savitch): For any p.c.f. (don't need this restric., actually) $f(n) \geq \log n$,

$$NSPACE(f(n)) \subseteq SPACE(f(n)^2).$$

Pf: Let M be $f(n)$-space NTM for $L \in NSPACE(f(n))$.
Let $|x|=n$ be input.

Config graph $G_{M,x}$ has $N = K^{f(n)}$ nodes; $x \in L$ iff C_{acc} reachable in $G_{M,x}$ from C_{start}.

Run our space-eff. REACH alg. on $G_{M,x}$; uses space $\log^2 N = (f(n) \log K)^2 = O(f(n)^2)$.

? input string is x, not $G_{M,x}$ ....

It's okay; $\log^2 n$ space alg only "looks at" G to check if two nodes x, y have $x \to y$ edge ($i=0$ case)

Given two configs C_1, C_2 in $V(G_{M,x})$, can decide easily if C_1 yields C_2 in one time step, by looking at C_1, C_2, x, & transit. rule of M (hardwired in).

Q: did we use properness?
Kind of - first compute value $K \cdot f(n)$, then run PATH on $(C_{start}, C_{accept}, (\log K) \cdot f(n))$.

(Here, like last time, can also try $1, 2, 3, \ldots$ & get rid of p.c.f. assumption.)

So we now know:

$NPSPACE = PSPACE$

(space hier.)

$NL \subseteq SPACE(\log^2 n) \subsetneq PSPACE.$

Next time:
- NL-completeness
  - ↳ log-space reductions
- $PSPACE$-completeness
  - ↳ usual poly-time reduc.
