Last time: started space complexity unit
REACH \in NC \quad \text{REACH} \in \text{SPACE}(\log n)^3

Savitch's Theorem: For any p.c.f. \( f(n) \geq \log n \), have \( \text{NSPACE}(f(n)) \subseteq \text{SPACE}(f(n)^3) \).

Today: log space reductions.

NL completeness \quad \text{TQBF}

REACH is NL-complete \quad \text{QSAT}

PSPACE completeness. (QBF is PSPACE-complete)

Admin: PS 2 due Thurs

Readings: AB 4.2, Papad. 19.1, Sipser 8.3, Cai 3.4

Questions?

---

Log-space reductions, NL-completeness

Q: is \( L = \text{NL} \)? probably... not?

Believed \( \text{REACH} \notin L \).

Can't answer this, but can simplify landscape: analog to NPC theory.

There are problems complete for NL: if anyone of them is in L, then \( \text{NL} = L \).

\text{REACH} is such a problem.
Can't use $\leq_p$ (poly-time red): too strong.

Right notion:

**Def:** $A$ is log-space reducible to $B$ (written $A \leq_L B$)

if there is a func. $f : \Sigma^* \rightarrow \Sigma^*$, computable in logspace, s.t. $\forall x \in \Sigma^*$,

$x \in A \iff f(x) \in B$

but must be $\leq \text{poly}(|x|)$ (b/c $f$ on length-$n$ input can only have $\leq \text{poly}(n)$ distinct configs, + can't repeat a config.)

**Def:** $B$ is $\text{NL}$-complete if

1. $B \in \text{NL}$, +
2. every $A \in \text{NL}$ has $A \leq_L B$.

**Fact:** If $A \leq_L B$ + $B \in \text{L}$, then $A \in \text{L}$.

**Pf:** Wrong arg: "on input $x \in A$, first use logspace-computable $f$ to compute $f(x)$, then run logspace alg for $B$ on $f(x)$".
Can’t do this: \( |f(x)| \) could be \( \gg \log n \).

Right approach: machine \( M_4 \) for \( A \) computes indiv. symbols of \( f(x) \) as required by \( M_B \) for its run on \( f(x) \).

\( \rightarrow \) (no writing \( f(x) \) explicitly)

\( M_4 \) simulates \( M_B \) on \( f(x) \) keeping track of loc. of its input head in \( f(x) \).

Every time \( M_B \) would move input tape head (say to loc. \( i \) in \( f(x) \)), \( M_4 \) restarts comp. of \( f \) on \( x \) (scratch tape) from beginning; ignores (doesn’t write down) all output of \( f \) except \( i \)th bit.) Since \( f \) (less space computable, i.e. poly \( n \) ), so \( O(\log n) \) space enough.

---

**Thm:** REACH is \( NL \)-complete.

**Pf:** Know (1) \( REACH \in NL \) (last time)

Need (2) every \( A \in NL \) has \( A \in REACH \).

Let \( A \in NL \), let \( M_A \) be \( \text{a losspace NTM for } A \).

Given \( x \in A \), construct \( (G,s,t) \) in losspace s.t.

\( G \) has \( s \to t \) path iff \( x \in A \).

\( G \) will be the config. graph of \( M_A \) on input \( x \):

\( G \) has \( s_{\text{start}} \to c_i \) path iff \( x \in A \).

nodes of \( G \): all poss. configs. of \( M_A \) of length \( \log n \).

edges: have \( c_i \to c_j \) iff \( M_A \), after one step from config. \( c_i \), can reach \( c_j \).

The machine which outputs config. graph \( G_{M_A,x} \) on input \( x \) in losspace:

- easy to output \( s_{\text{start}} \) and \( c_0 \)
outputs 6 as list of nodes, list of edges.

- nodes: easy (list all well-formed log(n) length strings, reusing space)
- edges: loop thru all pairs of nodes; for each candidate edge \( c_i \rightarrow c_j \), can verify whether \( M_k \) yields \( c_j \) from \( c_i \) in one time step in \( O(n) \) space.

PSPACE - completeness

Recall usual NPC SAT problem:

determining truth/falsity of

\[ \exists x_1 \exists x_2 \ldots \exists x_n \\forall (x_1, \ldots, x_n) \]

Generalization: consider expr. like

\[ \exists x, \forall x_2 \exists x_3, \forall x_4 \ldots \exists x_n \\forall (x_1, \ldots, x_n) \]

a totally quantified Boolean formula. Can capture e.g. \( \exists x, \exists x_2 \forall x_3 \): just ignore \( 1 \) (dummy var.).

Every var. in \( \varphi \) has some quantifier:

\[ \exists x: \forall x \exists y (x \lor y) \land (\neg x \lor y) \] is TRUE
\( \exists y \forall z (y \land z) \quad \text{FALSE} \)

\( \exists x \, (x \lor y) \quad \text{not tot. quantified} \)

**Taut:**

\( \forall x, \forall x_2 \ldots \forall x_n \, \phi(x, \ldots, x_n) \)

Def: Language

TQBF

\( \text{QSAT} = \{ \Phi : \Phi \text{ is a true tot. quant. Bool. form.} \} \)

\( \text{QSAT} \in \text{NP?} \quad \text{Seems... no?} \)

Seems "above whole \text{PH}"... (pays \text{BGS debt in full})

**Thm:** \( \text{QSAT is PSPACE-complete} : \)

1. \( \text{QSAT} \in \text{PSPACE,} \)
2. \( \forall \leq \text{PSPACE, have} \leq_p \text{QSAT.} \)

**Pf of (i):** here's poly space \( \Phi \) to det. whether an input \( \Phi \) is in \( \text{TQBF}: \)

**Alg 4:**

- check \( \Phi \) well-formed (all vars quantified)

So \( \Phi \) looks like

\( \exists x, \forall x, \exists x, \forall x \ldots \forall x_n \, \psi(x, \ldots, x_n) \)
- if $\Phi$ is $\exists x \alpha$: recursively call $A$ on
  $\alpha$ with $x$ replaced by $0x$
  $\alpha$ " " " " $T$ (reusing space for
  2nd call. If at least one of
  returns $T$, return $T$,
  else return $F$.
- if $\Phi$ is $\forall x \alpha$: recursively call $A$ on
  $\alpha$ with $x$ replaced by $0x$
  $\alpha$ " " " " $T$ (reusing space for
  2nd call. If both calls
  returns $T$, return $T$,
  else return $F$.
- if $\Phi$ has no quantifiers: all vars set to values,
  so just evaluate $\Phi$ and return result.

$A$ is correct. Depth of recursion = $n$
Space usage: $\text{poly}(n)$ overhead per level of recursion,
so $\text{poly}(n)$ overall.

Sketch of 2): $A \in \text{PSPACE}$, have $\text{LE} \subseteq \text{SAT}.$

Fix any $L \in \text{PSPACE}$.

Let $M$ be (det) TM running in $n^k$ space and
deciding $L$. (Say $M$ has 1 tape).
If $M$ accepts, always does so (and halts)
in time $2^{dn^k}$ some const. $d$, $n$ ($\omega(n^k)$, $\text{LE} = \text{P}$)

$\text{LE} \subseteq \text{P}$.
We need to give a polytime red. which, on input $x \in L$, output a QBF which is true iff $x \in L$.

3 ideas/ingredients:

1) Cook-Levin thm: computation tableau, cell $T_{i,j}$ contains contents of TM $M$'s tape cell $j$ at time $t$.
   As in CL pf, can use Bool formulas to enforce consistency of $T$ with $M$'s comput.
   **NOTE:** in $n^k$ space, $M$ runs for as long as $2^{d \cdot n^k}$ time, so tableau has $n^k$ columns, $2^{d \cdot n^k}$ rows.

2) Savitch's theorem: recursively find midpoint of tableau.
   This would require $2^{d \cdot n^k} - 1$ $\exists \ldots \exists$ quantifiers...

3) Use uni quantifiers to save (expon.) on length of formula.
   (enables poly size QBF to capture the computation)

Next time: - finish
- another PSPACE-complete problem
  (Geography)
- IM/SZ (clever use of nondet.)