## Non-deterministic TM (NTM)

extend $\delta: Q \times \Gamma^k \to \mathcal{P}(Q \times \Gamma^k \times \{\leftarrow, \rightarrow, 0\}^k)$ ... choice of instruction from the set

- successor relation is not a function → multiple computations for a given output
  - WLOG $|\delta(q,x)| \le 2$
- if $\delta(q,x) = \emptyset$, we assume rejection.
- input $\alpha$ accepted ≡ ∃ at least one accepting computation
- halting ≡ all computations halt
- time & space: maximum over all computations

enumeration works again ($NM_\alpha$ = NTM with code $\alpha$), we have an universal NTM &c.

Exercise: k-tape → 2-tape NTM with only constant-factor slowdown

Df: Non-deterministic complexity classes: NTIME($f$), NTIMEF($f$) for functions
└ all accepting computations must agree on result, ∃ at least one accepting comp.

Theorem: $NP = NTIME(poly(n)) := \bigcup_{k \ge 0} NTIME(n^k)$

Proof: $\subseteq$ the NTM guesses the certificate using non-determinism & then it runs the verifier
$\supseteq$ the certificate encodes ~~the~~ non-deterministic choices

Example: The following problem is NP-complete:
$\{\langle \alpha, \beta, 1^t \rangle \mid NM_\alpha \text{ accepts input } \beta \text{ within } t \text{ steps}\}$
→ ∈NP: simulate $NM_\alpha(\beta)$ using universal NTM with an "alarm clock" (reject after $t$ steps)
↓ reduction: calculate $t \in poly(n)$, $\alpha :=$ code of NTM solving source problem, pass ~~as~~ $\beta$

## Space Complexity

We want to count only "work space" of the TM.

3 types of tapes:
- input tape: read-only, head doesn't move more than 1 cell before/after input string
- k work tapes: read-write
- output tape: write-only, head cannot move left

} Space used by computation := # visited cells on the work tapes (for NTM: max over computations)

This doesn't change time complexity classes: we can copy input → work → output with constant slowdown

We can encode information in position of head on input tape.
- this makes a difference if work space $\in o(\log n)$
- otherwise we can keep track of the head position in binary

Space classes:

DSPACE($f$), NSPACE($f$) } decision problems
& DSPACEF($f$), NSPACEF($f$) } functions

We want $f$ to be:
1. non-decreasing
2. space-constructible
   ≡ $f(n)$ can be computed from $1^n$, result in binary in space $O(f(n))$

} proper space-complexity function

3. usually $f(n) \ge \log n$

PSPACE = DSPACE(poly($n$))
NPSPACE = NSPACE(poly($n$))

**Basic inclusions:** $\mathrm{DTIME}(f) \subseteq \mathrm{NTIME}(f) \subseteq \mathrm{DSPACE}(f) \subseteq \mathrm{NSPACE}(f)$

can try all certificates in space $O(f)$

So: $P \subseteq NP \subseteq PSPACE \subseteq NPSPACE$

**Thm:** $\mathrm{DSPACE}(f) \subseteq \mathrm{DTIME}(2^{O(f)})$ for every $f \geq \log n$.

**Proof:** First, let's bound # reachable configurations: $\underbrace{|Q|}_{\text{state}} \cdot \underbrace{(n+2)}_{\text{pos. of head on input tape}} \cdot \underbrace{(|\Gamma|+1)^{f(n)}}_{\text{contents of work tapes, extra character for "end of tape"}} \cdot \underbrace{f(n)^{k}}_{\text{pos. of heads on work tapes}}$ ← # tapes

... this is $O(2^{O(f(n))})$

If a configuration repeats, the whole computation loops. (⊛ this requires deterministic TM!)

⇒ add a binary counter of $O(f(n))$ bits, use it as alarm clock (increment in every step of the original TM). Alarm expires ⇒ reject.

in space-bounded computation with space $\geq \log n$, we can always make sure than the machine halts.

**Corollary:** ~~N~~PSPACE $\subseteq$ EXPTIME $:= \mathrm{DTIME}(2^{O(n)})$.

We want to prove the same for NSPACE(f), but ⊛ makes it more complicated.

**Reachability method**

**Df:** Configuration graph of a given NTM on a given input (within space $f(n)$) is a directed graph with:

- $V :=$ set of configurations (for input tape, consider only head position) ← limited by available space
- $E :=$ successor relation
- start $\in V$ ... initial config
- accept $\in V$ ... modify the TM to clean up before accepting < clear working tapes, rewind input tape } unique accepting config

- $|V| \in O(2^{O(f)})$, $|E| \in O(|V|)$, graph can be generated in $O(\mathrm{poly}(|V|))$ time & $O(f)$ space.
- Machine accepts ⇔ graph contains a (directed) path from start to accept.

**Thm:** $\mathrm{NSPACE}(f) \subseteq \mathrm{DTIME}(2^{O(f)})$ for every $f \geq \log n$. [therefore NPSPACE $\subseteq$ EXPTIME]

**Proof:** Construct the reachability graph & run BFS on it.

- construct: time $O(\mathrm{poly}(|V|, |E|))$ ← which is $O(2^{O(f)})$
- BFS: also time $O(\mathrm{poly}(|V|, |E|))$

**Generally:** Time-/space-efficient algorithms for REACH translate to inclusions of complexity classes.

REACH $= \{\langle G, s, t\rangle \mid \exists$ path from $s$ to $t$ in $G\}$

**Thm (Savitch's):** $\mathrm{NSPACE}(f) \subseteq \mathrm{DSPACE}(f^2)$ for every $f \geq \log n$.

**Corollary:** NPSPACE $\subseteq$ PSPACE, so NPSPACE = PSPACE.

→ will be proven soon...

**Lemma:** REACH $\in O(\log^2 n)$

**Proof:** Use "middle-first search".

Recursive function $D_k(x,y)$ computing "$\exists$ walk from $x$ to $y$ with at most $2^k$ edges".

We have: $D_0(x,y) = (x=y) \vee ((x,y)\in E)$

$D_k(x,y) = \bigvee_{z\in V} (D_{k-1}(x,z) \,\&\, D_{k-1}(z,y))$ ... FOR loop & recursion

Every level of recursion requires $O(\log n)$ space for local variables, $\log n$ levels suffice to find a path.

Now, we want to combine generator of config graph with this algorithm, but we don't have space to store the graph.

**Lemma:** If $f$ can be computed in space $s(n)$ and $g$ — " — $t(n)$, } $s,t \geq \log$

Then $g(f(\ldots))$ can be computed in space $O(s(n) + t(2^{O(s(n))}))$

} also applies to composition of a language with a function (that is, a reduction)

**Proof:**

input
↓
machine for $f$ (F)
↓ intermediate output of size $O(\text{time}(F)) \subseteq O(2^{O(s(n))})$ — denote by $m$
machine for $g$ (G)
↓
output

Start G & keep track of position of head on G's input tape. [$O(\log m)$ space]

Whenever G moves its input head (& at the start of computation), re-run F to get the corresponding symbol of its output.

↳ modify F: reset work tapes on startup
- reset input head — " —
- keep track of output head position ] $O(\log m)$ space
- write to output tape: compare with G's input head pos:
  - $=$ remember char in state
  - $\neq$ discard written char (won't be read by F)

Total space:
- $s(n)$ for F
- $O(\log m)$ for head positions ... this is $O(t(m))$
- $t(m)$ for G

**Corollary:** Savitch's thm.

↳ If $L \in$ NSPACE$(f)$: graph generation requires $O(f)$ space, reachability needs $O((2^{O(f)})^2) = O(2^{O(f)})$ ] combined by the lemma to $O(2^{O(f)})$

**Remark:** REACH $\in O(\log n)$ would imply NSPACE$(f)$ = DSPACE$(f)$ ... but this is long open.

It's known that undirected UREACH $\in O(\log n)$ [Reingold 2004, non-trivial]

↳ this implies only SSPACE$(f)$ = DSPACE$(f)$

↳ symmetric non-determinism (successor relation symmetric)

So we have: DTIME$(f) \subseteq$ NTIME$(f) \subseteq$ DSPACE$(f) \subseteq$ NSPACE$(f) \subseteq$ DTIME$(2^{O(f)})$

$\subseteq$ DSPACE$(f^2)$

and: NSPACE$(\log n) \subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE = NPSPACE $\subseteq$ EXPTIME

↑ this is also known as NL

↑ also = co-NPSPACE as PSPACE is closed under complements