LBA, Space complexity, *PSPACE*, *L*, and *NL*

Textbook: Chapter 5.1, 8

Finite TMs (LBA)

- TMs are definitionally *infinite*
- Reality is definitionally *finite*
- Corollary: TMs are impossible in real life!
- **We have to use a finite-sized tape**
  - Or a computer that can manufacture new RAM, but imagine a memory leak!

Formal definition of LBA

Def: A **linear bounded automaton** (LBA) is a restricted type of TM which has no memory outside of its input. Equivalently, it has some fixed constant amount of memory.

- An LBA configuration looks like $w_1 w_2 \ldots w_i q_j w_{i+1} \ldots w_k$, where k **is constant a given input**
- Since a configuration is of constant length, we can check if it has occurred before in the computation history!
- Therefore, the acceptance and halting problems for LBA are **decidable!**
- However, there are still **infinitely many inputs**: Therefore, E_{LBA} is still undecidable

Space complexity

- We measured time complexity to be the number of steps before halting
- In real computation, time is not the only important factor! Memory is often just as, if not more, expensive
- We can measure memory/space complexity in the same way

Def: Let M be a DTM that always halts. The **space complexity** of M on input of length n is the function $f : \mathcal{N} \to \mathcal{N}$, where $f(n)$ is the maximal number of tape cells that M scans. If the space complexity of M is $f(n)$, we say M runs in space $f(n)$.

If M is a **NTM**, the space complexity is the maximal number of tape cells scanned on **any single branch** of the nondeterminism.

Space complexity classes

- We usually use asymptotic notation
- Just like we had *TIME* and *NTIME*, we can define *SPACE* and *NSPACE*

Def: Space complexity class operators.

$$SPACE(f(n)) = \{L : L \texttt{ is decided by an } O(f(n)) \texttt{ space DTM}\}$$
$$NSPACE(f(n)) = \{L : L \texttt{ is decided by an } O(f(n)) \texttt{ space NTM}\}$$

Savitch's theorem

- We don't think $P = NP$, but does the space equivalent hold?
- As it turns out, **yes!** DTMs can simulate NTMs in polynomial *space*

Def: Savitch's theorem. For any function $f : \mathcal{N} \rightarrow \mathcal{R}^+$ where $f(n) \geq log(n)$,

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

Pf. excluded.

PSPACE

- Just like we have P for the set of all languages decidable in polynomial time, we can define $PSPACE$ for the set of all languages decidable in polynomial deterministic space

Def: $PSPACE$ is the set of all languages decidable in deterministic polynomial space.

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

$PSPACE = NPSPACE$

Savitch's theorem says that, for any natural number ϵ, $NSPACE(n^{\epsilon}) \subseteq SPACE(n^{2\epsilon})$. Therefore, $PSPACE = NPSPACE$.

- We know $PSPACE = NPSPACE$ by Savitch's theorem. Any problem $\in P$ can access at most polynomial space, and any problem $\in NP$ can access at most nondeterministic polynomial space ($NPSPACE$)
  - Therefore, $P \subseteq PSPACE = NPSPACE$ and $NP \subseteq PSPACE = NPSPACE$
- Let $EXPTIME$ be the set of all problems solvable in deterministic exponential time

Class subsets and *PSPACE*-completeness

$$P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXPTIME$$

- Researchers think that all the $\subseteq$ above are actually $\subsetneq$, but we don't know yet
- We know that at least one is proper

Def: A language B is $PSPACE$-**complete** if it satisfies the following two conditions:

1. $B \in PSPACE$, and
2. Every $A \in PSPACE$ is polynomial-time reducible to B

If only the second holds, we say B is $PSPACE$-**hard**.

L and NL

- Since $PSPACE = NPSPACE$, we need a new thing to care about
- What about logarithmic space?

Def: L is the class of languages decidable in logarithmic deterministic space. NL is the class of languages decidable in logarithmic nondeterministic space.

$$L = SPACE(\log n)$$
$$NL = NSPACE(\log n)$$

Next up: Intractability