Space Complexity
We measure space complexity by looking at the largest tape index reached during the computation.
Let $M$ be a deterministic TM.

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

**Definition:** $\text{SPACE}(S(n)) = \{ L \mid L \text{ is decided by a Turing machine with } O(S(n)) \text{ space complexity} \}$
Theorem: $3\text{SAT} \in \text{SPACE}(n)$

“Proof”: Try all possible assignments to the (at most $n$) variables in a formula of length $n$. This can be done in $O(n)$ space.

Theorem: $\text{NTIME}(t(n))$ is in $\text{SPACE}(t(n))$

“Proof”: Try all possible computation paths of $t(n)$ steps for an NTM on length-$n$ input. This can be done in $O(t(n))$ space.
The class $\text{SPACE}(s(n))$ formalizes the class of problems solvable by computers with \textit{bounded memory}.

Fundamental (Unanswered) Question: How does time relate to space, in computing?

$\text{SPACE}(n^2)$ problems could potentially take much longer than $n^2$ steps to solve

\textit{Intuition}: You can re-use space, but not time
Let $M$ be a halting TM that on input $x$, uses $S$ space

How many time steps can $M(x)$ possibly take?

Is there an upper bound?

The number of time steps is at most the total number of possible configurations!

(If a configuration repeats, the machine is looping.)

A configuration of $M$ specifies a head position, state, and $S$ cells of tape content. The total number of configurations is at most:

$$S |Q| |\Gamma|^S = 2^{O(S)}$$
Corollary:
Space $S(n)$ computations can be decided in $2^{O(S(n))}$ time:

$$\text{SPACE}(s(n)) \subseteq \bigcup_{c \in \mathbb{N}} \text{TIME}(2^c s(n))$$

Idea: After $2^{O(s(n))}$ time steps, a $s(n)$-space bounded computation must have repeated a configuration, so then it will never halt...
PSPACE = $\bigcup_{k \in \mathbb{N}} \text{SPACE}(n^k)$

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

PSPACE \subseteq \text{EXPTIME}
Is $P \subseteq \text{PSPACE}$?

YES
Is $\text{NP} \subseteq \text{PSPACE}$?

YES
Is $\text{NP}^{\text{NP}} \subseteq \text{PSPACE}$?

YES
P ⊆ NP ⊆ PSPACE ⊆ EXPTIME

Theorem: P ≠ EXPTIME

Why? The Time Hierarchy Theorem!

TIME(2^n) ∉ P

Therefore P ≠ EXPTIME
**Space Hierarchy Theorem**

**Intuition:** If you have more space to work with, then you can solve strictly more problems!

**Theorem:** For functions $s, S : \mathbb{N} \to \mathbb{N}$ where $s(n)/S(n) \to 0$

$\text{SPACE}(s(n)) \subsetneq \text{SPACE}(S(n))$

**Proof IDEA:** Diagonalization:
Make a machine $M$ that uses $S(n)$ space and “does the opposite” of all $s(n)$ space machines on at least one input

So $L(M)$ is in $\text{SPACE}(S(n))$ but not $\text{SPACE}(s(n))$
Nondeterministic Space Classes

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

The space complexity of $N$ is the function $f : N \to N$, where $f(n)$ is the furthest tape cell reached by $N$ over all computation paths, over all inputs of length $n$.

Definition: $\text{NSPACE}(s(n)) = \{ L \mid L \text{ is decided by a non-deterministic Turing Machine with } O(s(n)) \text{ space complexity} \}$
Savitch’s Theorem

Theorem: for every “nice” function \( s : \mathbb{N} \to \mathbb{N} \), where \( s(n) > n \) for all \( n \), \( \text{NSPACE}(s(n)) \subseteq \text{SPACE}(s(n)^2) \)

Open: does \( \text{NSPACE}(s(n)) = \text{SPACE}(s(n)) \)?

\[
\text{NPSPACE} = \bigcup_{k \in \mathbb{N}} \text{NSPACE}(n^k)
\]

Corollary: \( \text{NPSPACE} = \text{PSPACE} \)
Parting thoughts:
More Space, more power
Relation of time and space wide open.
Polynomial-time hierarchy contained in PSPACE, which contains interesting problems ("chess" is PSPACE complete). Non-determinism may reduce space but less dramatically.