Two different concepts:

$O$: no more than something

$o$: less than something

Definition

$$f(n) = o(g(n))$$

if

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0.$$
Small-o II

The definition of this limit:

\[ \forall c > 0, \exists n_0, \forall n \geq n_0, \frac{f(n)}{g(n)} \leq c. \]

\( O \) versus \( o \):

\[ \exists c > 0, \exists n_0, \forall n \geq n_0, f(n) \leq cg(n) \]
\[ \forall c > 0, \exists n_0, \forall n \geq n_0, f(n) \leq cg(n) \]

The \( \forall c \) causes \( o \) to be something smaller.
\[ \sqrt{n} = o(n) \]

\[
\lim_{n \to \infty} \frac{\sqrt{n}}{n} = \lim_{n \to \infty} \frac{1}{\sqrt{n}} = 0
\]

\[ f(n) \neq o(f(n)) \]

\[
\lim_{n \to \infty} \frac{f(n)}{f(n)} = 1 \neq 0
\]
Example: $A = \{0^k1^k \mid k \geq 0\}$

- Let’s count the number of steps in the algorithm discussed before
- Check if the input is

\[
0...01...1
\]

This takes $O(n)$

- Move back: $O(n)$
- Cross off each 0 and 1: $O(n)$

How many such crosses: $n/2$

\[
n/2 \times O(n) = O(n^2)
\]
Example: $A = \{0^k1^k \mid k \geq 0\}$

- Accept or not?
  - $O(n)$ to go through from beginning to end
- Total:
  
  $$O(n) + O(n^2) + O(n) = O(n^2)$$
Definition:

\[ \text{TIME}(t(n)) \equiv \{ L \mid L \text{ a language decided by an } O(t(n)) \text{ TM} \} \]

We have

\[ \{0^k1^k \mid k \geq 0\} \in \text{TIME}(n^2) \]

Can we make it faster?
New Algorithm for $A = \{0^k1^k \mid k \geq 0\}$

- The procedure: cross off every other 0 and 1
  
  0000011111
  0011
  01
  $\epsilon$

  key: length of the string left must be always even

- A failed algorithm
  
  000011
  001

- Algorithm
New Algorithm for $A = \{0^k1^k \mid k \geq 0\}$

1. check 0...0 1...1
2. repeat if not empty
   total # 0 & 1: odd $\Rightarrow$ reject
   cross off every other 0 and 1
3. no 0 & 1 remain, accept
   - If 13 “0” $\Rightarrow$ 6 “0” $\Rightarrow$ 3 “0” $\Rightarrow$ 1 “0”
     $1 + \log_2 n$ iterations
   - Each iter: $O(n)$ operations
   - Total cost: $O(n \log n)$
New Algorithm for $A = \{0^k1^k \mid k \geq 0\}$ III

- Therefore

\[ \{0^k1^k \mid k \geq 0\} \in \text{TIME}(n \log n) \]

- Can we do better? no

- Any language decided in $o(n \log n)$ on a single-tape TM $\Rightarrow$ regular (not proved here)

- But we know that

\[ \{0^k1^k \mid k \geq 0\} \]

is not regular
Using two-tape TM for \( \{0^k1^k \mid k \geq 0\} \)

- We can have an \( O(n) \) procedure
  1. check 0...0 1...1
  2. copy 0 to 2nd tape
  3. find the first 1
  4. sequentially cut 1 and 0
     - if no “0” reject
     - if “1” left, reject
  otherwise, accept

- Each step \( O(n) \)