# CMSC 250: Structural Mathematical Induction

Justin Wyss-Gallifent

April 20, 2023

# 1 Introduction

Weak and strong mathematical induction are both predicated on the fact that we are proving something for all $n \geq n_0$ for some $n_0$. This means that there is some organization of the items for $n = n_0$, $n = n_0 + 1$, $n = n_0 + 2$ and so on.

However not all collections of objects are organized like this. Some examples:

- The set of binary trees cannot necessarily be organized this way.
- A recursively defined set cannot necessarily be organized this way.

However both of these things are defined by giving some elements in the set (sort of like base cases) and then some rule(s) for adding new elements to the sets. Structural induction is essentially a way of doing induction on these recursively defined sets.

# 2 How it Works

Suppose we want to prove some property is true for all items in a recursively defined set. We proceed as follows:

(a) Base Cases(s): We prove that the property is true for the original items in the set.

(b) Inductive Step: We prove that when the rules add new things to the set that the property is preserved.

# 3 Examples

Here are a bunch of examples.

**Example 3.1.** Suppose a set $S$ is defined recursively as follows:

(a) $1 \in S$

(b) If $x \in S$ then $2x \in S$.

Let's prove that all elements of $S$ are integer powers of 2.

(a) Base Case: Observe that $1 = 2^0$ so 1 is an integer power of 2.

(b) Inductive Step: Suppose $x \in S$ and $x$ is an integer power of 2. We claim that $2x$ is an integer power of 2 as well. Since $x$ is a power of 2 we know $x = 2^k$ for some $k \in \mathbb{Z}$. Then $2x = 2(2^k) = 2^{k+1}$ and since $k + 1 \in \mathbb{Z}$ we know that $2x$ is also a power of 2.

Take a few minutes to see that the inductive step is showing that the property is preserved. Whenever the original item in $S$ has the property and we add a new items to $S$, that new item will also have the property. It follows that all items have that property.

**Example 3.2.** Suppose a set $S$ is defined recursively as follows:

(a) $(0,0) \in S$

(b) If $(x,y) \in S$ then $(x,y+1),(x+1,y+1),(x+2,y+1) \in S$.

Let's prove that every element $(x,y) \in S$ has $x \leq 2y$.

(a) Base Case: Observe that $(0,0)$ certainly satisfies $0 \leq 2(0)$.

(b) Inductive Step: Suppose $(x,y) \in S$ and $x \leq 2y$. We have three things to show since there are three rules for adding new elements:

- We need to prove that $(x,y+1)$ satisfies $x \leq 2(y+1)$. However since $x \leq 2y \leq 2y+2 = 2(y+1)$ this is true.
- We need to prove that $(x+1,y+1)$ satisfies $x+1 \leq 2(y+1)$. However since $x \leq 2y$ we have $x+1 \leq 2y+1 \leq 2y+2 = 2(y+1)$ this is true.
- We need to prove that $(x+2,y+1)$ satisfies $x+2 \leq 2(y+1)$. However since $x \leq 2y$ we have $x+2 \leq 2y+2 = 2(y+1)$ this is true.

**Example 3.3.** Suppose a set $S$ of strings of $a$ and $b$ is defined recursively as follows:

(a) The empty string is in $S$.

(b) If $x \in S$ then $axb, bxa$ are in $S$ and if $x,y \in S$ then $xy \in S$.

Let's prove that every string in $S$ has an equal number of $a$ and $b$.

(a) Base Case: Observe that the empty string has 0 of each, and $0 = 0$.

(b) Inductive Step: We have three things to show since there are three rules for adding new elements:

- We need to prove that if $x \in S$ has an equal number of $a$ and $b$ then $axb$ has an equal number of $a$ and $b$. But this is clear since we're adding one of each.
- We need to prove that if $x \in S$ has an equal number of $a$ and $b$ then $bxa$ has an equal number of $a$ and $b$. But this is clear since we're adding one of each.
- We need to prove that if $x,y \in S$ each has an equal number of $a$ and $b$ then $xy$ has an equal number of $a$ and $b$. But this is clear since if $x$ has $k$ of each and $y$ has $j$ of each then $xy$ has $j+k$ of each.

**Example 3.4.** Suppose a set $S$ is defined recursively as follows:

(a) $0 \in S$

(b) If $x \in S$ then $2x + 1 \in S$.

Let's prove that $S = \{2^n - 1 \,|\, n \in \mathbb{Z}^{\geq 0}\}$.

To prove this we actually need to prove both $\subseteq$ and $\supseteq$. The first of these is by structural induction, the second by weak induction.

Let's first show that $S \subseteq \{2^n - 1 \,|\, n \in \mathbb{Z}^{\geq 0}\}$. This means showing that everything in $S$ has the form $2^n - 1$ for some $n \in \mathbb{Z}^{\geq 0}$.

(a) Base Case: Observe that $0 = 2^0 - 1$ so it is true.

(b) Inductive Step: Suppose that $x \in S$ and $x = 2^n - 1$ for some $n \in \mathbb{Z}^{\geq 0}$. We claim that $2x - 1$ also has this form. However observe that: $2x - 1 = 2(2^n - 1) - 1 = 2^{n+1} - 1$ and since $n + 1 \in \mathbb{Z}^{\geq 0}$ we are done.

Let's next show that $\{2^n - 1 \,|\, n \in \mathbb{Z}^{\geq 0}\} \subseteq S$ using weak induction. This means proving that for all $n \geq 0$ we have $2^n - 1 \in S$.

(a) Base Case: We check $n = 0$. Since $2^0 - 1 = 0$ and we know $0 \in S$ was given, the base case is true.

(b) Inductive Step: Suppose $2^k - 1 \in S$ for $k \geq 0$. We claim $2^{k+1} - 1 \in S$. Well observe that since $2^k - 1 \in S$ we know that $2(2^k - 1) + 1 \in S$ by the construction of $S$. However:

$$2(2^k - 1) + 1 = 2^{k+1} - 2 + 1 = 2^{k+1} - 1$$

and so $2^{k+1} - 1 \in S$ as desired.

**Example 3.5.** Recall how binary trees are defined recursively. Let's prove that $N(T) = E(T) + 1$ for any binary tree $T$.

(a) Base Case: If $T$ is a single node then $N(T) = 1$ and $E(T) = 0$ and $1 = 0 + 1$ is true.

(b) Inductive Step: There are two ways to create new binary trees during the recursive construction and we must examine both of them.

- Suppose $T$ is a binary tree for which $N(T) = E(T)+1$ and we create a new binary tree $T'$ by creating a new root node and attaching $T$ to it. We claim that $N(T') = E(T') + 1$.

  In creating this new binary tree we add one node and one edge and so $N(T') = N(T) + 1$ and $E(T') = E(T) + 1$ and so then:

$$\begin{aligned} N(T') &= N(T) + 1 \\ &= E(T) + 1 + 1 \\ &= E'(T) + 1 \end{aligned}$$

- Suppose $T_1$ and $T_2$ are binary trees for which $N(T_1) = E(T_1) + 1$ and $N(T_2) = E(T_2) + 1$ and we create a new binary tree $T'$ by creating a new root node and attaching both $T_1$ and $T_2$ to it. We claim that $N(T') = E(T') + 1$.

  In creating this new binary tree we add one node and two edges and so $N(T') = N(T_1)+N(T_2)+1$ and $E(T') = E(T_1)+E(T_2)+2$ and so then:

$$\begin{aligned} N(T') &= N(T_1) + N(T_2) + 1 \\ &= E(T_1) + 1 + E(T_2) + 1 + 1 \\ &= E(T_1) + E(T_2) + 2 + 1 \\ &= E(T') + 1 \end{aligned}$$

**Example 3.6.** Let's prove that $L(T) \leq 2^{H(t)}$ for any binary tree $T$.

(a) Base Case: If $T$ is a single node then $L(T) = 1$ and $H(T) = 0$ and $1 \leq 2^0$ is true.

(b) Inductive Step: Again there are two things to show:

- Suppose $T$ is a binary tree for which $L(T) \leq 2^{H(T)}$. and we create a new binary tree $T'$ by creating a new root node and attaching $T$ to it. We claim that $L(T') \leq 2^{H(T')}$.

  In creating this new binary tree the number of leaves does not change and the height increases by 1, so $L(T') = L(T)$ and $H(T') = H(T) + 1$ and so then:

  $$L(T') = L(T) \leq 2^{H(T)} = 2^{H'(T)-1} \leq 2^{H'(T)}$$

- Suppose $T_1$ and $T_2$ are binary trees for which $L(T_1) \leq 2^{H(T_1)}$ and $L(T_2) \leq 2^{H(T_2)}$ and we create a new binary tree $T'$ by creating a new root node and attaching both $T_1$ and $T_2$ to it. We claim that $L(T') \leq 2^{H(T')}$.

  In creating this new binary tree the number of leaves in $T'$ is the sum of the number of leaves in $T_1$ and $T_2$, so:

  $$L(T') = L(T_1) + L(T_2)$$

  The height of $T'$ will be 1 plus the maximum of the heights of $T_1$ and $T_2$. However it's certainly the case that $H(T') \geq H(T_1) + 1$ and $H(T') \geq H(T_2) + 1$. Then we get:

  $$\begin{aligned} L(T') = L(T_1) + L(T_2) &\leq 2^{H(T_1)} + 2^{H(T_2)} \\ &\leq 2^{H(T')-1} + 2^{H(T')-1} \\ &\leq 2 \cdot 2^{H(T')-1} \\ &\leq 2^{H(T')} \end{aligned}$$

  Note: Another approach to managing the height is to first look at the case where $H(T_1) \geq H(T_2)$. In that case $H(T') = H(T_1) + 1$

and then:

$$
\begin{aligned}
L(T') = L(T_1) + L(T_2) &\leq 2^{H(T_1)} + 2^{H(T_2)} \\
&\leq 2^{H(T_1)} + 2^{H(T_1)} \\
&= 2^{H(T')-1} + 2^{H(T')-1} \\
&\leq 2 \cdot 2^{H(T')-1} \\
&\leq 2^{H(T')}
\end{aligned}
$$

The case where $H(T_2) \geq H(T_1)$ is exactly the same with the trees exchanged.

**Example 3.7.** Let's prove that $N(T) \leq 2^{H(T)+1} - 1$ for any binary tree $T$.

(a) Base Case: If $T$ is a single node then $N(T) = 1$ and $H(T) = 0$ and $1 \leq 2^{0+1} - 1$ is true.

(b) Inductive Step: Again there are two things to show:

- Suppose $T$ is a binary tree for which $N(T) \leq 2^{H(T)+1} - 1$. and we create a new binary tree $T'$ by creating a new root node and attaching $T$ to it. We claim that $N(T') \leq 2^{H(T')+1} - 1$. It's actually easier to show that $2^{H(T')+1} - 1 \geq N(T')$.

  In creating this new binary tree we add one node and the height increases by 1, so $N(T') = N(T) + 1$ and $H(T') = H(T) + 1$ and so then:

$$\begin{aligned} 2^{H(T')+1} - 1 &= 2^{H(T)+1+1} - 1 \\ &= 2 \cdot 2^{H(T)+1} - 1 \\ &\geq 2(N(T) + 1) - 1 \\ &= 2N(T) + 1 \\ &\geq N(T) + 1 \\ &= N(T') \end{aligned}$$

- Suppose $T_1$ and $T_2$ are binary trees for which $N(T_1) \leq 2^{H(T_1)+1} - 1$ and $N(T_2) \leq 2^{H(T_2)+1} - 1$ and we create a new binary tree $T'$ by creating a new root node and attaching both $T_1$ and $T_2$ to it. We claim that $N(T') \leq 2^{H(T')+1} - 1$.

  In creating this new binary tree the number of nodes in $T$ is the sum of the number of nodes in $T_1$ and $T_2$ plus 1 more so:

$$N(T') = N(T_1) + N(T_2) + 1$$

  The height is tricky. The height of $T'$ will be 1 plus the maximum of the heights of $T_1$ and $T_2$. However it's certainly the case that $H(T') \geq H(T_1) + 1$ and $H(T') \geq H(T_2) + 1$. Then we get:

$$\begin{aligned} N(T') = N(T_1) + N(T_2) + 1 &\leq 2^{H(T_1)+1} - 1 + 2^{H(T_2)+1} - 1 + 1 \\ &\leq 2^{H(T_1)+1} + 2^{H(T_2)+1} - 1 \\ &\leq 2^{H(T')} + 2^{H(T')} - 1 \\ &\leq 2^{H(T')+1} - 1 \end{aligned}$$