Metatheorems come in stylized form

For any $e, \xi, \phi, \rho, v, \xi', \text{ and } \rho'$ such that

$$\langle e, \xi, \phi, \rho \rangle \downarrow \langle v, \xi', \phi, \rho' \rangle,$$

*FACT*

Last time: “evaluation doesn’t change the set of global variables”
Metatheorems come in stylized form

For any $e, \xi, \phi, \rho, \nu, \xi', \rho'$ such that

\[ \langle e, \xi, \phi, \rho \rangle \downarrow \langle \nu, \xi', \phi, \rho' \rangle, \]

we have

\[ \text{dom } \xi = \text{dom } \xi'. \]

“Evaluation doesn’t change the set of global variables”
Metatheorems are proved by induction

Induction over structure (or height) of derivation trees $\mathcal{D}$

These are “math-class proofs” (*not* derivations)

Proof

- Has one case for each rule
- Has multiple cases for some syntactic forms
- Assumes the induction hypothesis for any proper sub-derivation (derivation of a premise)
- Template in book (and handout)
Assume the existence of a derivation

Could terminate in any rule!

Base case:

\[ D = \frac{\langle \text{LITERAL}(v), \xi, \phi, \rho \rangle \Downarrow \langle v, \xi, \phi, \rho \rangle}{\text{LITERAL}} \]

Both sides identical!

\[ \text{dom}\xi = \text{dom}\xi \]
Holds for formal-parameter lookup

Another base case:

\[
D = \frac{x \in \text{dom } \rho}{\langle \text{VAR}(x), \xi, \phi, \rho \rangle \Downarrow \langle \rho(x), \xi, \phi, \rho \rangle}
\]

Both sides identical!

\[\text{dom } \xi = \text{dom } \xi\]
Inductive case: good sub-derivation

Assignment to formal parameter

\[
\begin{aligned}
\mathcal{D} &= x \in \text{dom} \rho \\
&\quad \frac{\mathcal{D}_r}{\langle \text{SET}(x,e), \xi, \phi, \rho \rangle \downarrow \langle v, \xi', \phi, \rho' \rangle} \\
&\quad \text{FORMAL ASSIGN}
\end{aligned}
\]

By induction hypothesis on \( \mathcal{D}_r \), \( \text{dom} \xi = \text{dom} \xi' \)

Both sides have same domain!
Inductive case: good sub-derivation

True conditional

\[ \mathcal{D}_1 \]
\[ \langle e_1, \xi, \phi, \rho \rangle \Downarrow \langle v_1, \xi', \phi, \rho' \rangle \]
\[ v_1 \neq 0 \]

\[ \mathcal{D}_2 \]
\[ \langle e_2, \xi', \phi, \rho' \rangle \Downarrow \langle v_2, \xi'', \phi, \rho'' \rangle \]

\[ \langle \textnormal{IF}(e_1, e_2, e_3), \xi, \phi, \rho \rangle \Downarrow \langle v_2, \xi'', \phi, \rho'' \rangle \]

By induction hypothesis on \( \mathcal{D}_1 \), \( \text{dom} \xi = \text{dom} \xi' \)

By induction hypothesis on \( \mathcal{D}_2 \), \( \text{dom} \xi' = \text{dom} \xi'' \)

Therefore, both sides have same domain:
\( \text{dom} \xi = \text{dom} \xi'' \)
The only interesting case: assign to global

\[ \begin{array}{c}
\begin{array}{c}
\text{If } x \notin \text{dom } \rho \quad \text{and } x \in \text{dom } \xi \\
\end{array}
\end{array} \]

\[ \frac{\langle e, \xi, \phi, \rho \rangle \downarrow \langle v, \xi', \phi, \rho' \rangle}{\langle \text{SET}(x,e), \xi, \phi, \rho \rangle \downarrow \langle v, \xi' \{x \mapsto v\}, \phi, \rho' \rangle} \]

Do both sides have same domain?

- **Does** \( \text{dom } \xi = \text{dom}(\xi' \{x \mapsto v\}) \) ?

By induction hypothesis on \( D_r \), \( \text{dom } \xi = \text{dom } \xi' \)

And \( \text{dom}(\xi' \{x \mapsto v\}) = \text{dom } \xi' \cup \{x\} = \text{dom } \xi \cup \{x\} \)

But \( x \in \text{dom } \xi \)! So \( \text{dom } \xi \cup \{x\} = \text{dom } \xi \)
Next step: Judge this semantics

Bloom’s hierarchy
1. Remember
2. Understand
3. Apply
4. Analyze
5. Evaluate
6. Create

Two questions:
• What’s awkward
• What would help get closer to C/C++?
Your turn: Ideas?

Less awkward?

More C-like?
Location semantics

Simpler naming:

- Name stands for a mutable location
- Location holds values, updated with `set`
- Function is just another value
Evaluation judgment using store

Judgment $\langle e, \rho, \sigma \rangle \downarrow \langle v, \sigma' \rangle$

- Mappings in $\rho$ never change
- $\rho$ maps a name to a mutable location
- $\sigma$ is the store (contents of every location)

Intuition of the compiler writer:

- $\rho$ models the compiler’s “symbol table”
  “$x$ is in register 4”
- $\sigma$ models the contents of registers and memory

Classic semantic technique
Blank white card: One question

One question about operational semantics
And now, Scheme