Monday tute:

2pm in QUAD G031
INDUCTIVE DEFINITIONS
Preliminaries

We need a formal (meta-) language to describe and reason about properties of programming languages!
Preliminaries

We need a formal (meta-) language to describe and reason about properties of programming languages!

We need to specify things like the

- grammar
- scoping rules (static semantics)
- type system (static semantics)
- evaluation strategy (dynamic semantics)

of a language
We need a formal (meta-) language to describe and reason about properties of programming languages!

We need to specify things like the

- grammar
- scoping rules (static semantics)
- type system (static semantics)
- evaluation strategy (dynamic semantics)

of a language

Fortunately, it turns out that natural deduction (inference rules) can be used for all of these tasks!
Judgements and Inference Rules

Judgements: A judgement states that a certain property\(^a\) holds for an object, e.g.

- \(3 + 4 \times 5\) is a valid arithmetic expression
- the string "aba" is a palindrome
- 0.43423 is a floating point value

\(^a\) or a relationship between objects, but more on that later
Judgements and Inference Rules

Judgements: A judgement states that a certain property holds for an object, e.g.

- $3 + 4 \times 5$ is a valid arithmetic expression
- the string "$aba" is a palindrome
- $0.43423$ is a floating point value

We write

$$s \ A$$

...to express that a property $A$ holds for the object $s$ (or $s$ is in the set $A$)

---

*a* or a relationship between objects, but more on that later
For example:

- $6$ even
  - $6$ is even, or
  - $6$ is an element of the set of even numbers
- $3 + 4 \times 5$ expr
  - $3 + 4 \times 5$ is a syntactically correct expression, or
  - $3 + 4 \times 5$ an element of the set containing all syntactically correct expressions
- $0.43423$ float
  - ...
For example:
- 6 even
  - 6 is even, or
  - 6 is an element of the set of even numbers
- 3 + 4 * 5 expr
  - 3 + 4 * 5 is a syntactically correct expression, or
  - 3 + 4 * 5 an element of the set containing all syntactically correct expressions
- 0.43423 float
  - ...

Note:

→ Similar to predicates in Predicate Logic
→ The postfix notation will prove to be convenient later on, when the objects become fairly big
**Inference Rules**

are rules of the form:

If $J_1$, and $J_2$, and $\ldots$ and $J_n$ are **inferrable**, then $J$ is inferrable

\[
\frac{J_1 \quad J_2 \quad \ldots \quad J_n}{J}
\]

**Terminology:**

- $J_1, \ldots, J_n$ are called **premises**
- $J$ is called a **conclusion**
- if a rule has no premises, it is called an **Axiom**
Examples:

- Axiom “0 is a natural number”:
  \[
  0 \text{ nat}
  \]

- If \( n \) is a natural number, then \( n + 1 \) is a natural number as well:
  \[
  n \text{ nat} \\
  \hline
  n + 1 \text{ nat}
  \]
Even and odd numbers:
Even and odd numbers:

- Axiom: “0 is even”

\[
0 \text{ even}
\]
Even and odd numbers:

- Axiom: “0 is even”

\[
\begin{align*}
0 \text{ even} \\
\end{align*}
\]

- If \( x \) is even, then \( x + 2 \) is even

\[
\begin{align*}
x \text{ even} \\
x + 2 \text{ even}
\end{align*}
\]
Even and odd numbers:

→ Axiom: “0 is even”

\[
0 \text{ even}
\]

→ if \(x\) is even, then \(x + 2\) is even

\[
\frac{x \text{ even}}{x + 2 \text{ even}}
\]

→ if \(x\) is even, then \(x + 1\) is odd

\[
\frac{x \text{ even}}{x + 1 \text{ odd}}
\]
Even and odd numbers:

- Axiom: “0 is even”

\[
0 \text{ even}
\]

- If \( x \) is even, then \( x + 2 \) is even

\[
\frac{x \text{ even}}{x + 2 \text{ even}}
\]

- If \( x \) is even, then \( x + 1 \) is odd

\[
\frac{x \text{ even}}{x + 1 \text{ odd}}
\]

- If \( x \) is even and \( y \) is odd, then \( x + y \) is odd

\[
\frac{x \text{ even} \quad \text{odd } y}{x + y \text{ odd}}
\]
Inference rules which define a set can be used to show that a certain object is in the set.

- Find rule where conclusion matches statement which you want to prove
- Show that preconditions can be inferred
- Finished if rule is an axiom
How can grammars be expressed using inference rules?

Example:

A language $M$ of properly matched parenthesis:

$$M = \{ \epsilon, (), ()(), ()(), \ldots, (), (()()), \ldots, (), (), ()(), ()(), \ldots \}$$
HOW CAN GRAMMARS BE EXPRESSED USING INFERENCE RULES?

Example:

A language $M$ of properly matched parenthesis:

$$M = \{\epsilon, (), ()(), ()(), \ldots, (()), (((())), \ldots, ()(), ()()(), \ldots\}$$

How can this language be defined using natural language?
HOW CAN GRAMMARS BE EXPRESSED USING INFERENCE RULES?

Example:

A language $M$ of properly matched parenthesis:

$$M = \{ \epsilon, (), ()(), ()()(), \ldots, (), ((())), \ldots, ()(()) , ()()(), \ldots \}$$

How can this language be defined using natural language?

1. The empty string (denoted by $\epsilon$) is in $M$
2. If $s_1$ and $s_2$ are in $M$, then $s_1s_2$ is in $M$
3. If $s$ is in $M$, then $(s)$ is in $M$
How can this language be defined using EBNF?
How can this language be defined using EBNF?

→ EBNF for $M$:

$$M \rightarrow \epsilon \mid MM \mid (M)$$
How can it be defined using Inference Rules?

(1) The empty string (denoted by $\varepsilon$) is in $M$:

(2) If $s_1$ and $s_2$ are in $M$, then $s_1 s_2$ is in $M$

(3) If $s$ is in $M$, then $(s)$ is in $M$
How can it be defined using Inference Rules?

(1) The empty string (denoted by $\epsilon$) is in $M$:

$$
\frac{}{\epsilon \in M}
$$

(2) If $s_1$ and $s_2$ are in $M$, then $s_1s_2$ is in $M$

(3) If $s$ is in $M$, then $(s)$ is in $M$
How can it be defined using Inference Rules?

(1) The empty string (denoted by $\epsilon$) is in $M$:

\[
\frac{}{\epsilon \in M}
\]

(2) If $s_1$ and $s_2$ are in $M$, then $s_1 s_2$ is in $M$:

\[
\frac{s_1 \in M \quad s_2 \in M}{s_1 s_2 \in M}
\]

(3) If $s$ is in $M$, then $(s)$ is in $M$:
How can it be defined using Inference Rules?

(1) The empty string (denoted by $\epsilon$) is in $M$:

$$
\frac{}{\epsilon \ M}
$$

(2) If $s_1$ and $s_2$ are in $M$, then $s_1 s_2$ is in $M$

$$
\frac{s_1 \ M \ s_2 \ M}{s_1 s_2 \ M}
$$

(3) If $s$ is in $M$, then $(s)$ is in $M$

$$
\frac{s \ M}{(s) \ M}
$$
Can we show that \((())\) is in \(M\)?
Can we show that \( (()) \) is in \( M \)?

\[
\begin{align*}
(1) & \quad \frac{\epsilon M}{(()) M} \\
(3) & \quad \frac{() M}{(()) M} \\
(2) & \quad \frac{((() \quad M}{(()) M}
\end{align*}
\]
Can we show that \( ()(() \) is in \( M \)?

What happens if we start with rule (3)?
Would anything change if we added the rule:

\[
\frac{s \ M}{((s)) \ M}
\]

or

\[
\frac{(s) \ M}{s \ M}
\]

or

\[
\frac{(s) \ M}{s \ M}
\]

to the rule set?
ADMISSIBLE AND DERIVABLE RULES

\[
\frac{s \ M}{((s)) \ M}
\]

is derivable from previous rules
Admissible and Derivable Rules

\[
\frac{s \ M}{((s)) \ M}
\]

is derivable from previous rules

\[
\frac{(s) \ M}{s \ M}
\]

is admissible (since it does not change the language, but not derivable!)
**Admissible and Derivable Rules**

\[
\frac{s \, M}{((s)) \, M}
\]

is derivable from previous rules.

\[
\frac{() \, s \, M}{s \, M}
\]

is admissible (since it does not change the language, but not derivable!)

\[
\frac{(s) \, M}{s \, M}
\]

is not admissible, as it introduces new strings into the set.
IN \rightarrow \text{nat}
is the smallest set that is consistent with the rules.

Why the smallest set?

Objective: no junk. Only what must be in \( X \) shall be in \( X \).

Gives rise to a nice proof principle (rule induction)

Alternative (greatest set) occasionally also useful: coinduction
BACK TO SIMPLE EXAMPLE

\[
\begin{array}{c}
0 \text{ nat} \\
n \text{ nat} \implies n + 1 \text{ nat}
\end{array}
\]

→ nat is the set of natural numbers \(\mathbb{N}\).
→ But why not the set of real numbers? \(0 \in \mathbb{R}, n \in \mathbb{R} \implies n + 1 \in \mathbb{R}\).
→ \(\mathbb{N}\) is the smallest set that is consistent with the rules.
**Back to Simple Example**

\[
\begin{array}{rcc}
0 & \text{nat} & n \\[5pt]
\hline
n & \text{nat} & n + 1 & \text{nat}
\end{array}
\]

$\text{nat}$ is the set of natural numbers $\mathbb{N}$

- But why not the set of real numbers? $0 \in \mathbb{R}$, $n \in \mathbb{R} \implies n + 1 \in \mathbb{R}$

- $\mathbb{N}$ is the **smallest** set that is **consistent** with the rules.

**Why the smallest set?**

- Objective: **no junk**. Only what must be in $X$ shall be in $X$.
- Gives rise to a nice proof principle (rule induction)
- Alternative (greatest set) occasionally also useful: coinduction
Formally:

Rules \( a_1 X \quad \ldots \quad a_n X \) with \( a_1, \ldots, a_n, a \in A \)

define set \( X \subseteq A \)
Formally:

Rules \( a_1 X \ldots a_n X \) with \( a_1, \ldots, a_n, a \in A \)

define set \( X \subseteq A \)

Formally: set of rules \( R \subseteq A \) set \( \times A \) \( (R, X \) possibly infinite\)

Applying rules \( R \) to a set \( B \):
Formally:

Rules \( \frac{a_1 \ X \ \ldots \ \ X \ a_n}{a \ X} \) with \( a_1, \ldots, a_n, a \in A \)

define set \( X \subseteq A \)

Formally: set of rules \( R \subseteq A \text{ set } \times A \) \( (R, X \text{ possibly infinite}) \)

Applying rules \( R \) to a set \( B \):

\[ \hat{R}(B) \equiv \{x | \exists H. (H, x) \in R \land H \subseteq B\} \]

Example:
FORMALLY

Rules \( a_1 X \ldots a_n X \) with \( a_1, \ldots, a_n, a \in A \)

define set \( X \subseteq A \)

**Formally:** set of rules \( R \subseteq A \setminus \times A \) \((R, X \text{ possibly infinite})\)

**Applying rules** \( R \) to a set \( B \):

\[
\hat{R}(B) \equiv \{x \mid \exists H. (H, x) \in R \land H \subseteq B\}
\]

**Example:**

\[
R \equiv \{(\{\}, 0)\} \cup \{(\{n\}, n + 1)\}
\]

\[
\hat{R}(\{3, 6, 10\}) =
\]
Formally:

Rules \( \frac{a_1 \ X \ \ldots \ a_n \ X}{a \ X} \) with \( a_1, \ldots, a_n, a \in A \)

define set \( X \subseteq A \)

Formally: set of rules \( R \subseteq A \text{ set} \times A \) \((R, X \text{ possibly infinite})\)

Applying rules \( R \) to a set \( B \):

\[ \hat{R}(B) \equiv \{ x | \exists H. (H, x) \in R \land H \subseteq B \} \]

Example:

\[ R \equiv \{ (\{\}, 0) \} \cup \{ (\{n\}, n + 1) \} \]

\[ \hat{R}(\{3, 6, 10\}) = \{0, 4, 7, 11\} \]
The Set

**Definition:** \( B \) is \( R \)-closed iff \( \hat{R}(B) \subseteq B \)
Definition: \( B \) is \( R \)-closed iff \( \hat{R}(B) \subseteq B \)

Definition: \( X \) is the least \( R \)-closed subset of \( A \)

This does always exist:
THE SET

Definition: $B$ is $R$-closed iff $\hat{R}(B) \subseteq B$

Definition: $X$ is the least $R$-closed subset of $A$

This does always exist:

Facts: $B_1 \text{ } R\text{-closed} \land B_2 \text{ } R\text{-closed} \implies B_1 \cap B_2 \text{ } R\text{-closed}$
**The Set**

**Definition:** \( B \) is \( R \)-closed iff \( \hat{R}(B) \subseteq B \)

**Definition:** \( X \) is the least \( R \)-closed subset of \( A \)

This does always exist:

**Facts:** \( B_1 \ R \)-closed \( \land \ B_2 \ R \)-closed \( \implies \) \( B_1 \cap B_2 \ R \)-closed

\[ X = \bigcap\{ B \subseteq A. \ B \ R \text{-closed}\} \]
Generation from Above
GENERATION FROM ABOVE
GENERATION FROM ABOVE

A

$R$-closed

$R$-closed

$R$-closed
GENERATION FROM ABOVE

\[ A \]

\[ R \text{-closed} \]

\[ X \]

\[ R \text{-closed} \]

\[ R \text{-closed} \]
How to compute $X$?

$X$ as Fixpoint

Instead: view $X$ as least fixpoint, $X$ least set with $^R(X) = X$.

Fixpoints can be approximated by iteration:

$X_0 = ^R_0(fg)$

$X_1 = ^R_1(fg)$

$X_n = ^R_n(fg)$

$X! = S_{n+2}IN^R_n(fg) = X$.
How to compute $X$?
$X = \bigcap \{ B \subseteq A. B R \text{ -- closed} \}$ hard to work with.
Instead:
How to compute $X$?

$X = \bigcap\{B \subseteq A. B \ R \text{ is closed}\}$ hard to work with.

**Instead:** view $X$ as least fixpoint, $X$ least set with $\hat{R}(X) = X$. 
X AS FIXPOINT

How to compute $X$?

$X = \bigcap\{B \subseteq A. \ B \ R - \text{closed}\}$ hard to work with.

**Instead:** view $X$ as least fixpoint, $X$ least set with $\hat{R}(X) = X$.

Fixpoints can be approximated by iteration:

$$X_0 = \hat{R}^0(\{\}) = \{\}$$
**X as Fixpoint**

How to compute $X$?

$X = \cap \{ B \subseteq A. B \ R \text{ - closed} \}$ hard to work with.

Instead: view $X$ as least fixpoint, $X$ least set with $\hat{R}(X) = X$.

Fixpoints can be approximated by iteration:

$X_0 = \hat{R}^0(\{\}) = {}$

$X_1 = \hat{R}^1(\{\}) = $ rules without hypotheses

\[ \vdots \]
X AS FIXPOINT

How to compute $X$?

$X = \bigcap \{ B \subseteq A. B R - \text{closed} \}$ hard to work with.

**Instead:** view $X$ as least fixpoint, $X$ least set with $\hat{R}(X) = X$.

Fixpoints can be approximated by iteration:

$$X_0 = \hat{R}^0(\emptyset) = \emptyset$$
$$X_1 = \hat{R}^1(\emptyset) = \text{rules without hypotheses}$$
$$\vdots$$
$$X_n = \hat{R}^n(\emptyset)$$
How to compute $X$?

$X = \bigcap\{B \subseteq A. B \ R - \text{closed}\}$ hard to work with.

**Instead:** view $X$ as least fixpoint, $X$ least set with $\hat{R}(X) = X$.

**Fixpoints can be approximated by iteration:**

$$X_0 = \hat{R}^0(\{\}) = \{\}$$

$$X_1 = \hat{R}^1(\{\}) = \text{rules without hypotheses}$$

$$\vdots$$

$$X_n = \hat{R}^n(\{\})$$

$$X_\omega = \bigcup_{n \in \mathbb{N}} R^n(\{\}) = X$$
Generation from Below

$A$

$\hat{R}^0(\{\})$
Generation from Below

\[ \hat{R}^0(\{} \cup \hat{R}^1(\{} \]
\begin{equation*}
\hat{R}^0(\{\}) \cup \hat{R}^1(\{\}) \cup \hat{R}^2(\{})
\end{equation*}
Generation from Below

\[ \hat{R}^0(\{\}) \cup \hat{R}^1(\{\}) \cup \hat{R}^2(\{\}) \cup \ldots \]