A Programming Language Based on Classical Logic

William Lovas
(with Karl Crary)
Motivation

“It is very, very easy to design bad programming languages.”

(John Reynolds)

- Want to design good programming languages by building on logical foundations
- Today: explore one possibility, classical logic
Part 1: Proof theory boot camp
Logical foundations

- Curry-Howard correspondence

<table>
<thead>
<tr>
<th>Logic</th>
<th>Programming</th>
</tr>
</thead>
<tbody>
<tr>
<td>Propositions</td>
<td>Types</td>
</tr>
<tr>
<td>Proofs</td>
<td>Programs</td>
</tr>
<tr>
<td>Proof-checking</td>
<td>Type-checking</td>
</tr>
<tr>
<td>Simplification</td>
<td>Evaluation</td>
</tr>
</tbody>
</table>

Classical logic as a programming language?
- excluded middle, proof by contradiction, …
Curry-Howard correspondence

- Propositions as types, proofs as programs
Curry-Howard correspondence

- Propositions as types, proofs as programs

- **Q:** What is a proof of a proposition?
Curry-Howard correspondence

- Propositions as types, proofs as programs
- Q: What is a proof of a proposition?
- Q: What is a proposition?
Curry-Howard correspondence

- Propositions as types, proofs as programs

- **Q:** What is a proof of a proposition?
- **Q:** What is a proposition?
- **A:** Something that can be judged true.
  - e.g. “2 + 3 = 6”, or “it is raining”
Example

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$. 

Example

- **Proposition:** If $A \text{ and }$ either $B$ or $C$, then either $A \text{ and } B$, or else $A \text{ and } C$.

- **Conjunction** ("and")
Example

- Proposition: If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- Conjunction ("and")
- Disjunction ("or")
Example

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- Conjunction (“and”)
- Disjunction (“or”)
- Implication (“if … then …”)
Example

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- Conjunction ("and")
- Disjunction ("or")
- Implication ("if ... then ...")
- $A$, $B$, and $C$: whatever you like...
Proposition: If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- Conjunction (“and”)
- Disjunction (“or”)
- Implication (“if ... then ...”)
- $A, B, \text{ and } C$: whatever you like...

Symbolically: $A \land (B \lor C) \Rightarrow (A \land B) \lor (A \land C)$
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.
- **Proof:**
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  - Suppose $A$ and either $B$ or $C$: 
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B$ or $C)$)
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B$ or $C))$
    - suppose $B$: 
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B$ or $C)$)
    - suppose $B$:
      - have $A$ (since $A$ and [...]
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...]) and ($B$ or $C$)
    - suppose $B$:
      - have $A$ (since $A$ and [...])
      - have $B$ (by assumption)
Example Proof

Proposition: If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

Proof:

- Suppose $A$ and either $B$ or $C$:
  - have $B$ or $C$ (since [...] and $(B$ or $C)$)
  - suppose $B$:
    - have $A$ (since $A$ and [...] )
    - have $B$ (by assumption)
    - thus $A$ and $B$ (since we have both)
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B$ or $C))$
    - suppose $B$:
      - have $A$ (since $A$ and [...] )
      - have $B$ (by assumption)
      - thus $A$ and $B$ (since we have both)
      - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
**Example Proof**

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B$ or $C)$)
    - suppose $C$:
      - have $A$ (since $A$ and [...]}
      - have $C$ (by assumption)
      - thus $A$ and $C$ (since we have both)
      - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
Example Proof

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B \text{ or } C)$)
    - suppose $B$:
      - ...
      - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
    - suppose $C$:
      - ...
      - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
**Example Proof**

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B \text{ or } C)$)
    - suppose $B$:
      - ...
        - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
    - suppose $C$:
      - ...
        - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
    - thus either $A$ and $B$, or $A$ and $C$
Example Proof

Proposition: If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

Proof:

- Suppose $A$ and either $B$ or $C$:
  - have $B$ or $C$ (since [...] and $(B$ or $C)$)
  - suppose $B$:
    - ...
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
  - suppose $C$:
    - ...
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
- thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
Proof:

Suppose $A$ and either $B$ or $C$:
- have $B$ or $C$ (since $[...]$ and $(B$ or $C)$)
- suppose $B$:
  - have $A$ (since $A$ and $[...]$)
  - have $B$ (by assumption)
  - thus $A$ and $B$ (since we have both)
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
- suppose $C$:
  - ...
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
  - thus either $A$ and $B$, or $A$ and $C$

thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
A closer look…

- **Proof:**
  - Suppose $A$ and either $B$ or $C$:
    - have $B$ or $C$ (since [...] and $(B$ or $C)$
    - suppose $B$:
      - have $A$ (since $A$ and [...] )
      - have $B$ (by assumption)
      - thus $A$ and $B$ (since we have both)
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
    - suppose $C$:
      - ...
      - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
    - thus either $A$ and $B$, or $A$ and $C$
  - thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
Proof:

Suppose $A$ and either $B$ or $C$:

- have $B$ or $C$ (since [...] and ($B$ or $C$))
- suppose $B$:
  - have $A$ (since $A$ and [...])
  - have $B$ (by assumption)
  - thus $A$ and $B$ (since we have both)
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
- suppose $C$:
  - ...
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
- thus either $A$ and $B$, or $A$ and $C$

thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
A closer look...

Proof:

- Suppose $A$ and either $B$ or $C$:
  - have $B$ or $C$ (since [...] and $(B \lor C)$)
  - suppose $B$:
    - have $A$ (since $A$ and [...] )
    - have $B$ (by assumption)
    - thus $A$ and $B$ (since we have both)
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
  - suppose $C$:
    - ...
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
    - thus either $A$ and $B$, or $A$ and $C$
- thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
A closer look...

Proof:

- Suppose \( A \) and either \( B \) or \( C \):
  - have \( B \) or \( C \) (since [...] and (\( B \) or \( C \)));
  - suppose \( B \):
    - have \( A \) (since \( A \) and [...])
    - have \( B \) (by assumption)
    - thus \( A \) and \( B \) (since we have both)
    - thus either \( A \) and \( B \), or \( A \) and \( C \) (in particular, the first)
  - suppose \( C \):
    - ...
    - thus either \( A \) and \( B \), or \( A \) and \( C \) (in particular, the second)
    - thus either \( A \) and \( B \), or \( A \) and \( C \)
- thus, if \( A \) and either \( B \) or \( C \), then either \( A \) and \( B \), or \( A \) and \( C \)
Proof:

- Suppose \( A \) and either \( B \) or \( C \):
  - have \( B \) or \( C \) (since […] and (\( B \) or \( C \))
  - suppose \( B \):
    - have \( A \) (since \( A \) and […]
    - have \( B \) (by assumption)
    - thus \( A \) and \( B \) (since we have both)
    - thus either \( A \) and \( B \), or \( A \) and \( C \) (in particular, the first)
  - suppose \( C \):
    - …
    - thus either \( A \) and \( B \), or \( A \) and \( C \) (in particular, the second)
    - thus either \( A \) and \( B \), or \( A \) and \( C \)
- thus, if \( A \) and either \( B \) or \( C \), then either \( A \) and \( B \), or \( A \) and \( C \)
Proof:

- Suppose $A$ and either $B$ or $C$:
  - have $B$ or $C$ (since $[...]$ and $(B$ or $C)$)
  - suppose $B$:
    - have $A$ (since $A$ and $[...]$)
    - have $B$ (by assumption)
    - thus $A$ and $B$ (since we have both)
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
  - suppose $C$:
    - ...
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
- thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
Proof:

Suppose $A$ and either $B$ or $C$:
- have $B$ or $C$ (since $[...]$ and $(B \lor C)$)
- suppose $B$:
  - have $A$ (since $A$ and $[...]$)
  - have $B$ (by assumption)
  - thus $A$ and $B$ (since we have both)
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
- suppose $C$:
  - $\ldots$
  - thus either $A$ and $B$, or $A$ and $C$
- thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
Formalizing Proof, take 1

- Judgement: $A \text{ true}$. ("$A$ is provable.")
- Inference rules: grouped into “Introductions”:

\[
\frac{A \text{ true} \quad B \text{ true}}{A \land B \text{ true}} \quad \wedge\text{-I}
\]

- … and “Eliminations”:

\[
\frac{A \land B \text{ true}}{A \text{ true}} \quad \wedge\text{-E}_1 \quad \frac{A \land B \text{ true}}{B \text{ true}} \quad \wedge\text{-E}_2
\]
A closer look…

Proof:

- Suppose $A$ and either $B$ or $C$:
  - have $B$ or $C$ (since $[...]$ and $(B$ or $C)$)
  - suppose $B$:
    - have $A$ (since $A$ and $[...]$)
    - have $B$ (by assumption)
    - thus $A$ and $B$ (since we have both)
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
  - suppose $C$:
    - ...
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
- thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
A closer look...

Proof:

- Suppose **A and either B or C:**
  - have B or C  
    - (since [...] and (B or C))
  - suppose B:
    - have A  
      - (since A and [...])
    - have B  
      - (by assumption)
    - thus A and B  
      - (since we have both)
  - thus either A and B, or A and C  
    - (in particular, the first)
  - suppose C:
    - ...
    - thus either A and B, or A and C  
      - (in particular, the second)
    - thus either **A and B, or A and C**
  - thus, if A and either B or C, then either A and B, or A and C
A closer look…

**Proof:**

- Suppose $A$ and either $B$ or $C$:
  - have $B$ or $C$ (since [...] and $(B$ or $C)$
  - suppose $B$:
    - have $A$ (since $A$ and [...])
    - have $B$ (by assumption)
    - thus $A$ and $B$ (since we have both)
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
  - suppose $C$:
    - ...
    - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
  - thus either $A$ and $B$, or $A$ and $C$

- thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
A closer look...

Proof:

Suppose $A$ and either $B$ or $C$:

- have $B$ or $C$ (since $[...]$ and $(B$ or $C)$)
- suppose $B$:
  - have $A$ (since $A$ and $[...]$)
  - have $B$ (by assumption)
  - thus $A$ and $B$ (since we have both)
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
- suppose $C$:
  - ...
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
  - thus either $A$ and $B$, or $A$ and $C$

thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
Proof:

Suppose $A$ and either $B$ or $C$:

- have $B$ or $C$ (since [...] and [...]
- suppose $B$:
  - have $A$ (since $A$ and [...] and [...]
  - have $B$ (by assumption)
  - thus $A$ and $B$ (since we have both)
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the first)
- suppose $C$:
  - ...
  - thus either $A$ and $B$, or $A$ and $C$ (in particular, the second)
- thus either $A$ and $B$, or $A$ and $C$

thus, if $A$ and either $B$ or $C$, then either $A$ and $B$, or $A$ and $C$
Formalizing Proof, take 2

- Judgement: $\Gamma \vdash A \text{ true}$. ("$A$ is provable assuming $\Gamma$")
- $\Gamma$ is a list of assumptions: $A_1 \text{ true}, \ldots, A_n \text{ true}$
- Implication: one introduction rule:

$$
\frac{\Gamma, A \text{ true} \vdash B \text{ true}}{\Gamma \vdash A \Rightarrow B \text{ true}} \Rightarrow \text{-I}
$$
Formalizing Proof, take 2

- Judgement: $\Gamma \vdash A \text{ true.}$ ("$A$ is provable assuming $\Gamma$")
- $\Gamma$ is a list of assumptions: $A_1 \text{ true, } \ldots, A_n \text{ true}$
- Implication: one introduction rule:

$$
\frac{\Gamma, A \text{ true} \vdash B \text{ true}}{\Gamma \vdash A \Rightarrow B \text{ true}} \Rightarrow \text{-I}
$$

- … and one elimination rule:

$$
\frac{\Gamma \vdash A \Rightarrow B \text{ true} \quad \Gamma \vdash A \text{ true}}{\Gamma \vdash B \text{ true}} \Rightarrow \text{-E}
$$
Reasoning from assumptions

- Hypothesis rule:

\[ \Gamma, A \text{ true} \vdash A \text{ true} \]
Reasoning from assumptions

- Hypothesis rule:
  \[ \Gamma, A \text{ true} \vdash A \text{ true} \]

- Substitution Principle: if \( \Gamma, A \text{ true} \vdash B \text{ true} \) and \( \Gamma \vdash A \text{ true} \), then \( \Gamma \vdash B \text{ true} \)
Reasoning from assumptions

- Hypothesis rule:

\[ \Gamma, A \text{ true} \vdash A \text{ true} \]

- Substitution Principle: if \( \Gamma, A \text{ true} \vdash B \text{ true} \) and \( \Gamma \vdash A \text{ true} \), then \( \Gamma \vdash B \text{ true} \)
Reasoning from assumptions

- Hypothesis rule:

\[ \Gamma, A \text{ true} \vdash A \text{ true} \]

- Substitution Principle: if \( \Gamma, A \text{ true} \vdash B \text{ true} \) and \( \Gamma \vdash A \text{ true} \), then \( \Gamma \vdash B \text{ true} \)
Reasoning from assumptions

- Hypothesis rule:
  \[ \Gamma, A \text{ true} \vdash A \text{ true} \]

- Substitution Principle: if \( \Gamma, A \text{ true} \vdash B \text{ true} \) and \( \Gamma \vdash A \text{ true} \), then \( \Gamma \vdash B \text{ true} \)
Disjunction: two introduction rules:

\[
\frac{\Gamma \vdash A \text{ true}}{\Gamma \vdash A \lor B \text{ true}} \quad \lor\text{-I}_1 \\
\frac{\Gamma \vdash B \text{ true}}{\Gamma \vdash A \lor B \text{ true}} \quad \lor\text{-I}_2
\]
Formalizing Proof, take 3

Disjunction: two introduction rules:

\[
\begin{align*}
\text{\Gamma \vdash A \text{ true}} & \quad \therefore \text{\Gamma \vdash A \lor B \text{ true}} \\
\text{\Gamma \vdash B \text{ true}} & \quad \therefore \text{\Gamma \vdash A \lor B \text{ true}}
\end{align*}
\]

\[-I_1 \quad \text{\Gamma, A true} \vdash \text{true} \quad \therefore \text{\Gamma, A \lor \text{true} \vdash \text{true}} \quad \therefore \text{\Gamma, B \lor \text{true} \vdash \text{true}} \quad \therefore \text{\Gamma, A \lor B \lor \text{true} \vdash \text{true}}
\]

\[-I_2 \quad \text{\Gamma, B true} \vdash \text{true} \quad \therefore \text{\Gamma, A \lor B \lor \text{true} \vdash \text{true}}
\]

... and one elimination rule:

\[
\begin{align*}
\text{\Gamma \vdash A \lor B \text{ true}} & \quad \text{\Gamma, A true} \vdash \text{C true} & \quad \text{\Gamma, B true} \vdash \text{C true} \\
\therefore \text{\Gamma \vdash C \text{ true}} & \quad \therefore \text{\Gamma \vdash C \text{ true}} & \quad \therefore \text{\Gamma \vdash C \text{ true}}
\end{align*}
\]

\[-E \quad \text{\Gamma \vdash C \text{ true}}
\]
Proof simplification

- Easy to make detours. Consider proving $A \Rightarrow A$: 

Proof simplification

- Easy to make detours. Consider proving $A \Rightarrow A$:
- Suppose $A$ true:
Proof simplification

- Easy to make detours. Consider proving $A \Rightarrow A$:
  - Suppose $A$ true:
    - Hmm… tricky…
Proof simplification

- Easy to make detours. Consider proving $A \Rightarrow A$:

- Suppose $A$ true:
  - Hmm... tricky...
  - Well, we also have $B$ true...
Proof simplification

- Easy to make detours. Consider proving $A \Rightarrow A$:

- Suppose $A$ true:
  - Hmm… tricky…
  - Well, we also have $B$ true…
  - A-ha! By $\land$-I, we have $A \land B$ true.
Proof simplification

- Easy to make detours. Consider proving $A \Rightarrow A$:

- Suppose $A$ true:
  - Hmm… tricky…
  - Well, we also have $B$ true…
  - A-ha! By $\land$-I, we have $A \land B$ true.
  - And then by $\land$-E$_1$, we have $A$ true.
Proof simplification

- Easy to make detours. Consider proving $A \Rightarrow A$:

- Suppose $A$ true:
  - Hmm… tricky…
  - Well, we also have $B$ true…
  - A-ha! By $\land$-I, we have $A \land B$ true.
  - And then by $\land$-E₁, we have $A$ true.
  - *phew*
Proof simplification

- Eliminate “redundant” steps

\[ \vdash A \text{ true} \quad \Gamma \vdash B \text{ true} \]
\[ \Gamma \vdash A \land B \text{ true} \]
\[ \Gamma \vdash A \text{ true} \]
Proof simplification

- Eliminate “redundant” steps

\[
\frac{
D \\
\Gamma \vdash A \text{ true} \quad \Gamma \vdash B \text{ true}
}{
\Gamma \vdash A \land B \text{ true}
}
\]

\[
\frac{
\Gamma \vdash A \text{ true}
}{
\Gamma \vdash A \text{ true}
}
\]
Proof simplification

Using Substitution Principle:

\[
\begin{align*}
\text{If} & : \\
\Gamma, A \text{true} \vdash B \text{true} & \\
\hline
\Gamma \vdash A \Rightarrow B \text{true} & \quad \Gamma \vdash A \text{true} \\
\hline
\Gamma \vdash B \text{true}
\end{align*}
\]
Proof simplification

- Using Substitution Principle:

\[ \frac{\Gamma, A \ true \vdash B \ true}{\Gamma \vdash A \Rightarrow B \ true} \]

\[ \frac{\Gamma \vdash A \ true}{\Gamma \vdash B \ true} \]
Proof terms

\[
\Gamma \vdash A \text{ true} \quad \Gamma \vdash B \text{ true}
\]
\[\frac{}{\Gamma \vdash A \land B \text{ true}} \quad \land\text{-I}
\]

\[
\Gamma \vdash A \land B \text{ true}
\]
\[\frac{}{\Gamma \vdash A \text{ true}} \quad \land\text{-E}_1
\]
\[\frac{}{\Gamma \vdash B \text{ true}} \quad \land\text{-E}_2
\]
Proof terms

\[
\frac{\Gamma \vdash e_1 : A \; true \quad \Gamma \vdash e_2 : B \; true}{\Gamma \vdash (e_1, e_2) : A \land B \; true} \quad ^\bot \text{-I}
\]

\[
\frac{\Gamma \vdash A \land B \; true}{\Gamma \vdash A \; true} \quad ^\bot \text{-E}_1
\]

\[
\frac{\Gamma \vdash A \land B \; true}{\Gamma \vdash B \; true} \quad ^\bot \text{-E}_2
\]
Proof terms

\[ \Gamma \vdash e_1 : A \text{ true} \quad \Gamma \vdash e_2 : B \text{ true} \]
\[ \Gamma \vdash (e_1, e_2) : A \land B \text{ true} \quad \land \text{-I} \]

\[ \Gamma \vdash e : A \land B \text{ true} \]
\[ \Gamma \vdash \#1 e : A \text{ true} \quad \land \text{-E}_1 \]
\[ \Gamma \vdash \#2 e : B \text{ true} \quad \land \text{-E}_2 \]
Proof terms

- Conjunction: pairing!

\[ \Gamma \vdash e_1 : A \text{ true} \quad \Gamma \vdash e_2 : B \text{ true} \]
\[ \Gamma \vdash (e_1, e_2) : A \land B \text{ true} \]

\[ \frac{\Gamma \vdash e : A \land B \text{ true}}{\Gamma \vdash \#1 e : A \text{ true}} \quad \quad \frac{\Gamma \vdash e : A \land B \text{ true}}{\Gamma \vdash \#2 e : B \text{ true}} \]
Proof terms

\[
\begin{align*}
\Gamma, x : A \ true &\vdash e : B \ true \\
\Gamma &\vdash \lambda x. \ e : A \Rightarrow B \ true
\end{align*}
\Rightarrow -I
\]

\[
\begin{align*}
\Gamma &\vdash e_1 : A \Rightarrow B \ true \quad \Gamma &\vdash e_2 : A \ true \\
\Gamma &\vdash e_1 \ e_2 : B \ true
\end{align*}
\Rightarrow -E
\]
Proof terms

- Implication: functions!
- (Note: assumptions now labelled)

\[ \begin{align*}
\Gamma, x : A \text{ true} & \vdash e : B \text{ true} \\
\Gamma & \vdash \lambda x . e : A \Rightarrow B \text{ true} \\
\Gamma & \vdash e_1 e_2 : B \text{ true}
\end{align*} \]

\[ \Rightarrow \text{-I} \]

\[ \Rightarrow \text{-E} \]
Proof terms
Proof terms

- Disjunction: datatypes and pattern matching!
Proof terms

- Disjunction: datatypes and pattern matching!
Proof terms

- Disjunction: datatypes and pattern matching!
- (rules elided)
Proof terms

- Simplification: evaluation!

\[ \#1 (e_1, e_2) \rightarrow e_1 \]
\[ \#2 (e_1, e_2) \rightarrow e_2 \]
\[ (\lambda x. e_1) e_2 \rightarrow [e_2/x] e_1 \]

- Basic programming language: the simply-type lambda calculus.
  - data structures, functions
Example Proof, revisited
Example Proof, revisited

- Proposition: If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$. 
Example Proof, revisited

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.
- **Proof:**
Example Proof, revisited

- **Proposition:** If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

- **Proof:**
  
  $\text{fn } x : A \land (B \lor C) \Rightarrow$
  
  $\text{case } #2 \ x \ of \ \text{inl } y \Rightarrow \text{inl } (#1 \ x, y)$
  
  $\mid \text{inr } z \Rightarrow \text{inr } (#1 \ x, z)$
Example Proof, revisited

Proposition: If $A$ and either $B$ or $C$, then either $A$ and $B$, or else $A$ and $C$.

Proof:

\[ \text{fn } x : A \land (B \lor C) \Rightarrow \]

\[ \text{case } #2 x \text{ of } \text{inl } y \Rightarrow \text{inl } (#1 x, y) \]

\[ \mid \text{inr } z \Rightarrow \text{inr } (#1 x, z) \]

Computational content of proof: a simple input-shuffling program
Classical Logic

- What I’ve shown you: intuitionistic logic
- Classical logic: proof-by-contradiction

\[
\begin{align*}
\Gamma, A & \text{false} \vdash \text{contra} \\
\Gamma & \vdash A \text{true} \\
\Gamma & \vdash C \text{true} \\
\Gamma & \vdash C \text{false} \\
\Gamma & \vdash \text{contra}
\end{align*}
\]

- What is \textit{contra}?
- What is \textit{false}?
- Computational interpretation?
Continuations
Continuations

- Intuition: separate a program into what’s happening now and what happens next...
  - what’s happening now: expression currently being evaluated
  - what happens next: the continuation: the rest of the program
Current continuation

"letcc u in e": bind current continuation to u, run e

"throw e to u": restore continuation u with expr. e
  ◦ like a goto with an argument
Example: early exit

- letcc example: early exit
Example: early exit

- letcc example: early exit

```haskell
fun product [] = 1
  | product (x::xs) = x * product xs
```
Example: early exit

- letcc example: early exit

```haskell
fun product [] = 1
    | product (0::_) = 0
    | product (x::xs) = x * product xs
```
Example: early exit

- `letcc` example: early exit

```haskell
fun product nums =  
    letcc u in  
        let fun prod [] = 1  
            | prod (0::_) = throw 0 to u  
            | prod (x::xs) = x * prod xs  
        in  
            prod nums  
    end
```
What *are* continuations?

- Like a "partial program": given a value of the right type, it becomes a complete program.
- "A cont": type of a continuation expecting an A
What *are* continuations?

- Like a “partial program”: given a value of the right type, it becomes a complete program.
- “A cont”: type of a continuation expecting an $A$

in “early exit” example: $u : \text{int cont}$, since “product” should return an $\text{int}$

```
fun product nums =
  letcc u in
  let fun prod ...
  in
  prod nums
  end
```
What can we *do* with them?

- Given an *A cont*, pass it an *A*
What can we *do* with them?

- Given an $A$ cont, pass it an $A$
- Given an $A$ cont, construct an $A \land B$ cont
  - accept a pair : $A \land B$
  - project the first component : $A$
  - pass it to original continuation
What can we *do* with them?

- Given an $A \text{ cont}$, pass it an $A$
- Given an $A \text{ cont}$, construct an $A \land B \text{ cont}$
  - accept a pair : $A \land B$
  - project the first component : $A$
  - pass it to original continuation

\[
A \text{ cont} \Rightarrow A \land B \text{ cont}
\]
\[
B \text{ cont} \Rightarrow A \land B \text{ cont}
\]
What can we do with them?

- Given an \( A \) cont, pass it an \( A \)
- Given an \( A \) cont, construct an \( A \land B \) cont
  - accept a pair : \( A \land B \)
  - project the first component : \( A \)
  - pass it to original continuation
- Given an \( A \) cont and a \( B \) cont, make an \( A \lor B \) cont
  - accept a sum : \( A \lor B \)
  - case analyze it to get either an \( A \) or a \( B \)
  - if \( A \), pass to the \( A \) cont; if \( B \), pass to the \( B \) cont
What can we do with them?

- Given an $A$ cont, pass it an $A$
- Given an $A$ cont, construct an $A \land B$ cont
  - accept a pair : $A \land B$
  - project the first component : $A$
  - pass it to original continuation
- Given an $A$ cont and a $B$ cont, make an $A \lor B$ cont
  - accept a sum : $A \lor B$
  - case analyze it to get either an $A$ or a $B$
  - if $A$, pass to the $A$ cont; if $B$, pass to the $B$ cont
What can we do with them?

- Given an $A$ cont, pass it an $A$
- Given an $A$ cont, construct an $A \land B$ cont
  - accept a pair : $A \land B$
  - project the first component : $A$
  - pass it to original continuation
- Given an $A$ cont and a $B$ cont, make an $A \lor B$ cont
  - accept a sum : $A \lor B$
  - case analyze it to get either an $A$ or a $B$
  - if $A$, pass to the $A$ cont; if $B$, pass to the $B$ cont
Continuations are refutations!

\[
\begin{align*}
\Gamma \vdash k : A & \text{false} \\
\hline
\Gamma \vdash (#1; k) : A \land B & \text{false} & \land\text{-F}_1 \\
\Gamma \vdash k : B & \text{false} \\
\hline
\Gamma \vdash (#2; k) : A \land B & \text{false} & \land\text{-F}_2 \\
\Gamma \vdash k_1 : A & \text{false} & \Gamma \vdash k_2 : B & \text{false} \\
\hline
\Gamma \vdash [k_1, k_2] : A \lor B & \text{false} & \lor\text{-F}
\end{align*}
\]
Classical Curry-Howard

- Expressions: \( e : A \ true \)
- Continuations: \( k : A \ false \)
Classical Curry-Howard

- Expressions: \( e : A \ true \)
- Continuations: \( k : A \ false \)
- Programs: \( k \leftarrow e \)

\[ \Gamma \vdash e : C \ true \quad \Gamma \vdash k : C \ false \]
\[ \Gamma \vdash k \leftarrow e : contra \]
Classical Curry-Howard

- Evaluate programs \( k \triangleleft e \):

  1. \#1; \( k \triangleleft (e_1, e_2) \) \( \Rightarrow \) \( k \triangleleft e_1 \)
  2. \#2; \( k \triangleleft (e_1, e_2) \) \( \Rightarrow \) \( k \triangleleft e_2 \)
  3. \[k_1, k_2]\triangleleft \text{inl } e \Rightarrow k_1 \triangleleft e
  4. \[k_1, k_2]\triangleleft \text{inr } e \Rightarrow k_2 \triangleleft e
Proof-by-contradiction redux

\[ \Gamma, A \text{false} \vdash \text{contra} \]
\[ \Gamma \vdash A \text{true} \]
Proof-by-contradiction redux

\[
\Gamma, u : A \quad false \vdash k \triangleleft e : contra
\]

\[
\Gamma \vdash \text{letcc } u \text{ in } k \triangleleft e : A \; true
\]
Proof-by-contradiction redux

\[
\begin{align*}
\Gamma, u : A & \vdash k \triangleleft e : contra \\
\Gamma & \vdash \text{letcc } u \text{ in } k \triangleleft e : A \ true
\end{align*}
\]

\[
\begin{align*}
k' \triangleleft \text{letcc } u \text{ in } k \triangleleft e & \quad \rightarrow \quad [k'/u] (k \triangleleft e)
\end{align*}
\]
Classical proof terms

\[ e ::= x \mid \text{letcc } u:A \text{ false in } c \) \\
| (e_1, e_2) \mid () \\
| \text{inl}(e) \mid \text{inr}(e) \\
| \lambda x:A. e \\
| \text{not}(k) \]

\[ k ::= u \mid \text{let } x:A \text{ true in } c \]
| #1; k \mid #2; k \\
| [k_1, k_2] \mid [] \\
| e; k \\
| \text{not}(e) \]

\[ A \text{ true} \]
| A \land B, \top \\
| A \lor B \\
| A \Rightarrow B \\
| \neg A \\

\[ A \text{ false} \]
| A \land B \\
| A \lor B, \bot \\
| A \Rightarrow B \\
| \neg A \]
Normalization

- **Theorem:** Every contradiction has a normal form.
  - “normal”: cannot reduce any further

- **Proof:** By nested induction on the type at which a contradiction occurs and the terms undergoing evaluation.
**Normalization**

- **Theorem**: Every contradiction has a normal form.
  - “normal”: cannot reduce any further

- **Proof**: By nested induction on the *type* at which a contradiction occurs and the *terms* undergoing evaluation.

- **Corollary**: Classical logic is consistent, since there are no closed, normal contradictions
Prior work

- Standing on many giants’ shoulders:
  - Andrzej Filinski
  - Michel Parigot
  - Timothy Griffin
  - Chetan Murthy
  - Pierre-Louis Curien and Hugo Herbelin
  - Aleksandar Nanevski
  - Philip Wadler

- But one of the first -- and simplest -- proofs of normalization.
Conclusion

- Observed that continuations embody refutations of propositions
- Constructed a programming language with continuations, based on proof-by-contradiction
- Proved the language terminating, establishing the consistency of classical logic

(John Reynolds approves)
Q: What is a proof of a proposition A?
A: Depends on A…

How about $A \land B$?

A proof of $A \land B$ is a proof of $A$ and a proof of $B$.

\[
\frac{A \text{ true} \quad B \text{ true}}{A \land B \text{ true}}
\]
Q: What can we do with a proof?
A: From a proof of $A \land B$, we can get a proof of $A$.
(Also, a proof of $B$.)
Reasoning from hypotheses

- Refine judgement: $A \text{ true}$ becomes $\Gamma \vdash A \text{ true}$, with $\Gamma$ an unordered list of hypotheses $A_1, \ldots, A_n$

- Hypothesis rule:

  $\Gamma, A \text{ true} \vdash A \text{ true}$

- Substitution Principle: if $\Gamma, A \text{ true} \vdash B \text{ true}$ and $\Gamma \vdash A \text{ true}$, then $\Gamma \vdash B \text{ true}$
Intros and Elims

- Introduction

\[ \Gamma \vdash A \text{ true} \quad \Gamma \vdash B \text{ true} \]
\[ \Gamma \vdash A \land B \text{ true} \]

- Elimination

\[ \Gamma \vdash A \land B \text{ true} \]
\[ \Gamma \vdash A \text{ true} \]

\[ \Gamma \vdash A \land B \text{ true} \]
\[ \Gamma \vdash B \text{ true} \]
Proof simplification

- Eliminate “redundant” steps

\[
\frac{\Gamma \vdash A \ true \quad \Gamma \vdash B \ true}{\Gamma \vdash A \land B \ true} \quad \frac{\Gamma \vdash A \ true}{\Gamma \vdash A \ true}
\]
Implication

- Q: What is a proof of $A \rightarrow B$?
  - A: A proof of $B$, conditioned on a proof of $A$.

- Q: What can you do with a proof of $A \rightarrow B$?
  - A: Given a proof of $A$, make a proof of $B$. 
Implication rules

- A little more interesting…

\[ \Gamma, A \text{ true} \vdash B \text{ true} \]

\[ \Gamma \vdash A \Rightarrow B \text{ true} \]

\[ \Gamma \vdash A \Rightarrow B \text{ true} \quad \Gamma \vdash A \text{ true} \]

\[ \Gamma \vdash B \text{ true} \]
Using Substitution Principle:

\[
\begin{align*}
\Gamma, A \text{ true } & \vdash B \text{ true } \\
\Gamma & \vdash A \Rightarrow B \text{ true } & \Gamma & \vdash A \text{ true } \\
\Gamma & \vdash B \text{ true } & \Gamma & \vdash A \text{ true } \\
\Gamma & \vdash A \Rightarrow B
\end{align*}
\]
Proof terms

- Compact representation of derivations

\[
\Gamma \vdash M : A \text{ true} \quad \Gamma \vdash N : B \text{ true}
\]

\[
\Gamma \vdash (M, N) : A \land B \text{ true}
\]

\[
\Gamma \vdash M : A \land B \text{ true}
\]

\[
\Gamma \vdash \pi_1 M : A \text{ true}
\]

\[
\Gamma \vdash M : A \land B \text{ true}
\]

\[
\Gamma \vdash \pi_2 M : B \text{ true}
\]
Proof terms

- Hypothesis get labels: now $\Gamma$ is $x_1: A_1, \ldots, x_n: A_n$

- Hypothesis rule:

  $\Gamma, x: A \text{ true} \vdash x: A \text{ true}$

- Substitution Principle: if $\Gamma, x: A \text{ true} \vdash M : B \text{ true}$ and $\Gamma \vdash N : A \text{ true}$, then $\Gamma \vdash [N/x] M : B \text{ true}$
Proof terms

- Abstraction and application

\[
\Gamma, x : A \text{ true} \vdash M : B \text{ true} \\
\hline
\Gamma \vdash \lambda x : A \ M \cdot A \rightarrow B \text{ true} \\
\hline
\Gamma \vdash M : A \rightarrow B \text{ true} \quad \Gamma \vdash N : A \text{ true} \\
\hline
\Gamma \vdash MN : B \text{ true}
\]
Proof term simplification

- Reduction on trees $\Rightarrow$ reduction on terms

$$\pi_1 (M, N) \Rightarrow M$$

$$\pi_2 (M, N) \Rightarrow N$$

$$(\lambda x : A. M) N \Rightarrow [N/x] M$$

- This is a *programming language!*
Classical proof terms

\[ e ::= x \mid \text{letcc}(u \div A. \ c) \mid (e_1, e_2) \mid () \mid \text{inl}(e) \mid \text{inr}(e) \mid \lambda x : A. \ e \mid \text{not}(k) \]

\[ k ::= u \mid \text{let}(x : A. \ c) \mid k \circ \pi_1 \mid k \circ \pi_2 \mid [k_1, k_2] \mid [] \mid e; k \mid \text{not}(e) \]

\text{true}:
- \( A \land B, \top \)
- \( A \lor B \)
- \( A \Rightarrow B \)
- \( \neg A \)

\text{false}:
- \( A \land B \)
- \( A \lor B, \bot \)
- \( A \Rightarrow B \)
- \( \neg A \)