An Introduction to the Logic of Partial Functions

Matthew Lovert

Newcastle University

May 21, 2010
Outline

1. Partial Functions

2. Approaches
   - Changing the Notion of Equality
   - McCarthy’s Conditional Operators
   - The Logic of Partial Functions
Partial functions and operators arise frequently:
  - Division
  - Recursive function definitions
  - ...

Classical logic does not handle undefined terms:
  - Terms must be denoting
The subp Function

\[ \text{subp} : \mathbb{Z} \to \mathbb{Z} \to \mathbb{Z} \]

\[ \text{subp}(i, j) \triangleq \text{if } i = j \text{ then } 0 \text{ else } \text{subp}(i, j + 1) + 1 \]

Property 1

\[ \forall i, j \in \mathbb{Z} \cdot i \geq j \Rightarrow \text{subp}(i, j) = i - j \]

<table>
<thead>
<tr>
<th>( \Rightarrow )</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>
How do you handle undefined terms?

John Harrison:

Four main approaches to handling undefined terms:
- Give a value for input outside of the domain
- Arbitrary value
- Type error
- Logic of partial terms

We will now consider three approaches in more detail.
Changing the Notion of Equality

- Weak (strict) equality as we have seen is undefined if either operand is undefined.
- Alternative: Provide non-strict versions of equality which will denote a Boolean value even with undefined terms as operands.

### Weak equality

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>✏️</th>
<th>⊥️</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>true</td>
<td>false</td>
<td>✏️</td>
<td>⊥️</td>
</tr>
<tr>
<td>1</td>
<td>false</td>
<td>true</td>
<td>✏️</td>
<td>⊥️</td>
</tr>
<tr>
<td>✏️</td>
<td>✏️</td>
<td>✏️</td>
<td>✏️</td>
<td>✏️</td>
</tr>
<tr>
<td>⊥️</td>
<td>⊥️</td>
<td>⊥️</td>
<td>✏️</td>
<td>⊥️</td>
</tr>
</tbody>
</table>

### Existential equality

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>✏️</th>
<th>⊥️</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>true</td>
<td>false</td>
<td>✏️</td>
<td>false</td>
</tr>
<tr>
<td>1</td>
<td>false</td>
<td>true</td>
<td>✏️</td>
<td>false</td>
</tr>
<tr>
<td>✏️</td>
<td>✏️</td>
<td>✏️</td>
<td>✏️</td>
<td>✏️</td>
</tr>
<tr>
<td>⊥️</td>
<td>false</td>
<td>false</td>
<td>✏️</td>
<td>false</td>
</tr>
</tbody>
</table>
Changing the Notion of Equality Continued...

Property 1

\[ \forall i, j \in \mathbb{Z} \cdot i \geq j \Rightarrow \text{subp}(i, j) = \exists i - j \]

- The non-strict equalities are not computable.
- Not just required for equality.
- Need to be aware of more than one notion of the operators in proofs:
  - Strict (computable) operators in function definitions
  - Non-strict operators to handle undefined terms
McCarthy’s Conditional Operators

Interpret the propositional operators as if they are defined by conditional expressions:

- $p \text{ cor } q \mid \text{ if } p \text{ then } true \text{ else } q$
- $p \text{ cimp } q \mid \text{ if } p \text{ then } q \text{ else } true$
- ...

Loss of commutativity (cand and cor), and contrapositive of cimp.

Inevitable variable

Alternative: Use alongside the standard logical operators:

- Distribution is now complicated
McCarthy’s Conditional Operators Continued...

- \( p \, \texttt{cimp} \, q \mid \text{if } p \text{ then } q \text{ else } \text{true} \)

Property 1

\[ \forall i, j \in \mathbb{Z} \cdot i \geq j \, \texttt{cimp} \, \texttt{subp}(i, j) = i - j \]

- \( p \, \texttt{cor} \, q \mid \text{if } p \text{ then } \text{true} \text{ else } q \)

Property 2

\[ \forall i, j \in \mathbb{Z} \cdot \texttt{subp}(i, j) = i - j \, \texttt{cor} \, \texttt{subp}(j, i) = j - i \]
The Logic of Partial Functions

- Vienna Development Method (VDM)
- A first-order predicate logic, which admits to undefined logical terms.
- Three-valued logic: \( \text{true} \), \( \text{false} \), and \( \text{undefined} \) (\( \bot \)).
- The truth tables are the strongest extension of their classical FOPC interpretations.

<table>
<thead>
<tr>
<th>( \lor )</th>
<th>( \text{true} )</th>
<th>( \bot_B )</th>
<th>( \text{false} )</th>
<th>( \Rightarrow )</th>
<th>( \text{true} )</th>
<th>( \bot_B )</th>
<th>( \text{false} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{true} )</td>
<td>( \text{true} )</td>
<td>( \text{true} )</td>
<td>( \text{true} )</td>
<td>( \text{true} )</td>
<td>( \text{true} )</td>
<td>( \bot_B )</td>
<td>( \text{false} )</td>
</tr>
<tr>
<td>( \bot_B )</td>
<td>( \text{true} )</td>
<td>( \bot_B )</td>
<td>( \bot_B )</td>
<td>( \bot_B )</td>
<td>( \bot_B )</td>
<td>( \text{false} )</td>
<td>( \bot_B )</td>
</tr>
<tr>
<td>( \text{false} )</td>
<td>( \text{true} )</td>
<td>( \bot_B )</td>
<td>( \text{false} )</td>
<td>( \text{false} )</td>
<td>( \text{true} )</td>
<td>( \text{true} )</td>
<td>( \text{true} )</td>
</tr>
</tbody>
</table>

- Parallel evaluation of the operands.
- Return a result as soon as enough information is available.
The Logic of Partial Functions Continued...

- No law of the excluded middle ($e \lor \neg e$):
  - $\text{subp}(0, 5) = -5 \lor \neg (\text{subp}(0, 5) = -5)$
- Definedness operator ($\delta$):
  - $\delta(e) = e \lor \neg e$

**Property 1**

$$\forall i, j \in \mathbb{Z} \cdot i \geq j \Rightarrow \text{subp}(i, j) = i - j$$

**Property 2**

$$\forall i, j \in \mathbb{Z} \cdot \text{subp}(i, j) = i - j \lor \text{subp}(j, i) = j - i$$

- Mechanisation
J. H. Cheng and C. B. Jones

*On the usability of logics which handle partial functions.*


C. B. Jones.

*Reasoning about partial functions in the formal development of programs.*

Thank you.
Any Questions?