Constructive Proofs and Program Extraction

1. Type Theory vs. Set Theory
2. Overview of the Nuprl System
3. Proofs of the Integer Square Root Problem
What distinguishes Type Theory from Set Theory?

What is the meaning of \( \forall n \exists r \ r^2 \leq n \land n < (r+1)^2 \)?
What distinguishes Type Theory from Set Theory?

What is the meaning of $\forall n \exists r \, r^2 \leq n \land n < (r+1)^2$?

- **Set Theory**
  - Integer square roots exist for all natural numbers
What distinguishes Type Theory from Set Theory?

What is the meaning of $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Set Theory**
  - Integer square roots exist for all natural numbers
  - A proof shows that the notion $\lfloor \sqrt{n} \rfloor$ is well-defined
What distinguishes Type Theory from Set Theory?

What is the meaning of \( \forall n \exists r \ r^2 \leq n \land n < (r+1)^2 \)?

- Set Theory
  - Integer square roots exist for all natural numbers
  - A proof shows that the notion \( \lfloor \sqrt{n} \rfloor \) is well-defined
  - Different proofs lead to the same insight
What distinguishes Type Theory from Set Theory?

What is the meaning of $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$?

- **Set Theory**
  - Integer square roots exist for all natural numbers
  - A proof shows that the notion $\lfloor \sqrt{n} \rfloor$ is well-defined
  - Different proofs lead to the same insight

- **Type Theory**
  - There is a method to construct integer square roots
What distinguishes Type Theory from Set Theory?

What is the meaning of $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$?

- **Set Theory**
  - Integer square roots exist for all natural numbers
  - A proof shows that the notion $\lfloor \sqrt{n} \rfloor$ is well-defined
  - Different proofs lead to the same insight

- **Type Theory**
  - There is a method to construct integer square roots
  - Algorithms for computing $\lfloor \sqrt{n} \rfloor$ can be extracted from proofs
What distinguishes Type Theory from Set Theory?

What is the meaning of $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$?

- **Set Theory**
  - Integer square roots exist for all natural numbers
  - A proof shows that the notion $\lfloor \sqrt{n} \rfloor$ is well-defined
  - Different proofs lead to the same insight

- **Type Theory**
  - There is a method to construct integer square roots
  - Algorithms for computing $\lfloor \sqrt{n} \rfloor$ can be extracted from proofs
  - Different proofs lead to different algorithms
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- Nonconstructive
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min \{ r \mid r^2 > n \}$-1
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction

Unnecessary indirect approach - it masks a constructive argument
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction

  Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor:** *Induction on* $n$
Proofs for $\forall n \exists r \; r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction
    Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor: Induction on $n$**
  - Base Case: prove $\exists r \; r^2 \leq 0 \land 0 < (r+1)^2$
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction
  
  Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor: Induction on $n$**
  - **Base Case**: prove $\exists r \ r^2 \leq 0 \land 0 < (r+1)^2$
    
    Choose $r = 0$, prove $0^2 \leq 0 \land 0 < (0+1)^2$ using standard arithmetic
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r | r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction
  Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor: Induction on $n$**
  - Base Case: prove $\exists r \ r^2 \leq 0 \land 0 < (r+1)^2$
    Choose $r = 0$, prove $0^2 \leq 0 \land 0 < (0+1)^2$ using standard arithmetic
  - Step case: assume $\exists r_n \ r^2 \leq n \land n < (r_n+1)^2$ and
    prove $\exists r \ r^2 \leq n+1 \land n+1 < (r+1)^2$
PROOFS FOR $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction

Unnecessary indirect approach – it masks a constructive argument

- **Mathematical rigor: Induction on $n$**
  - **Base Case**: prove $\exists r \ r^2 \leq 0 \land 0 < (r+1)^2$
    - Choose $r = 0$, prove $0^2 \leq 0 \land 0 < (0+1)^2$ using standard arithmetic
  - **Step case**: assume $\exists r_n \ r^2 \leq n \land n < (r_n+1)^2$ and
    - prove $\exists r \ r^2 \leq n+1 \land n+1 < (r+1)^2$
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\}$-1
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction

  Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor: Induction on $n$**
  - Base Case: prove $\exists r \ r^2 \leq 0 \land 0 < (r+1)^2$
    Choose $r = 0$, prove $0^2 \leq 0 \land 0 < (0+1)^2$ using standard arithmetic
  - Step case: assume $\exists r_n \ r^2 \leq n \land n < (r_n+1)^2$ and
    prove $\exists r \ r^2 \leq n+1 \land n+1 < (r+1)^2$
  - If $(r_n+1)^2 \leq n+1$
    then choose $r = r_n+1$
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r \mid r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction

Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor: Induction on $n$**
  - Base Case: prove $\exists r \ r^2 \leq 0 \land 0 < (r+1)^2$
    Choose $r = 0$, prove $0^2 \leq 0 \land 0 < (0+1)^2$ using standard arithmetic
  - Step case: assume $\exists r_n \ r^2 \leq n \land n < (r_n+1)^2$ and
    prove $\exists r \ r^2 \leq n+1 \land n+1 < (r+1)^2$
    - If $(r_n+1)^2 \leq n+1$
      then choose $r = r_n+1$
    - otherwise choose $r = r_n$
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min \{ r \mid r^2 > n \}$ - 1
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction

  Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor: Induction on $n$**
  - **Base Case**: prove $\exists r \ r^2 \leq 0 \land 0 < (r+1)^2$
    Choose $r = 0$, prove $0^2 \leq 0 \land 0 < (0+1)^2$ using standard arithmetic
  - **Step case**: assume $\exists r_n \ r^2 \leq n \land n < (r_n+1)^2$ and
    prove $\exists r \ r^2 \leq n+1 \land n+1 < (r+1)^2$
    - If $(r_n+1)^2 \leq n+1$
      then choose $r = r_n+1$
    - Otherwise choose $r = r_n$
    - Proof obligation follow using standard arithmetic
Proofs for $\forall n \exists r \ r^2 \leq n \land n < (r+1)^2$

- **Nonconstructive**
  - Assume $r$ wouldn’t exist for some $n$
  - Then for all $r$: $r^2 > n$ or $(r+1)^2 \leq n$
  - Consider $r_1 = \min\{r | r^2 > n\} - 1$
  - Then $r_1^2 \leq n$ and $(r_1+1)^2 > n$ – a contradiction

Unnecessary indirect approach - it masks a constructive argument

- **Mathematical rigor:** Induction on $n$
  - Base Case: prove $\exists r \ r^2 \leq 0 \land 0 < (r+1)^2$
    
    Choose $r = 0$, prove $0 \leq 0 \land 0 < (0+1)^2$ using standard arithmetic
  
  - Step case: assume $\exists r_n \ r^2 \leq n \land n < (r_n+1)^2$ and
    
    prove $\exists r \ r^2 \leq n+1 \land n+1 < (r+1)^2$
  
  - If $(r_n+1)^2 \leq n+1$
    
    then choose $r = r_n + 1$
  
  - otherwise choose $r = r_n$

  - Proof obligation follow using standard arithmetic
  
  - Proof leads to algorithm that constructs $\lfloor \sqrt{n} \rfloor$ inductively
Constructive Proofsand Program Extraction

How to extract algorithms from proofs?

- Use formal logic to express proof
  - First-Order Logic + Induction + Basic Arithmetic $\subseteq$ Type Theory
  - Proof rules tie proof steps to algorithm fragments
**How to extract algorithms from proofs?**

- **Use formal logic to express proof**
  - First-Order Logic + Induction + Basic Arithmetic $\subseteq$ **Type Theory**
  - Proof rules tie proof steps to algorithm fragments

- **Use computerized proof assistant to formalize proof**
  - **Nuprl** proof development system supports formal proofs in type theory
  - Use proof tactics to keep formalization “simple”
HOW TO EXTRACT ALGORITHMS FROM PROOFS?

- Use **formal logic** to express proof
  - First-Order Logic + Induction + Basic Arithmetic $\subseteq$ Type Theory
  - Proof rules tie proof steps to algorithm fragments

- Use **computerized proof assistant** to formalize proof
  - **Nuprl** proof development system supports formal proofs in type theory
  - Use proof tactics to keep formalization “simple”

- **Extract algorithm** from computerized proof
  - Nuprl composes algorithm fragments of rules used in proof
  - Algorithm can be executed in Nuprl
The Nuprl System

Proof & program refinement in Type Theory

• Interactive Proof Editor \(\leadsto\) readable proofs
The Nuprl System

Proof & program refinement in Type Theory

- Interactive Proof Editor \(\leadsto\) readable proofs
- Proof Tactics \(\leadsto\) user-defined inferences
The Nuprl System

Proof & program refinement in Type Theory

- Interactive **Proof Editor** → readable proofs
- Proof **Tactics** → user-defined inferences
- **Decision Procedures** → proof automation
The Nuprl System

Proof & program refinement in Type Theory

- Interactive Proof Editor \( \leadsto \) readable proofs
- Proof Tactics \( \leadsto \) user-defined inferences
- Decision Procedures \( \leadsto \) proof automation
- Flexible definition mechanism \( \leadsto \) user-defined terms
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interactive <strong>Proof Editor</strong></td>
<td>readable proofs</td>
</tr>
<tr>
<td><strong>Proof Tactics</strong></td>
<td>user-defined inferences</td>
</tr>
<tr>
<td><strong>Decision Procedures</strong></td>
<td>proof automation</td>
</tr>
<tr>
<td><strong>Flexible definition mechanism</strong></td>
<td>user-defined terms</td>
</tr>
<tr>
<td><strong>Customizable Term Display</strong></td>
<td>flexible notation</td>
</tr>
</tbody>
</table>
The Nuprl System

Proof & program refinement in Type Theory

- Interactive Proof Editor \(\rightsarrow\) readable proofs
- Proof Tactics \(\rightsarrow\) user-defined inferences
- Decision Procedures \(\rightsarrow\) proof automation
- Flexible definition mechanism \(\rightsarrow\) user-defined terms
- Customizable Term Display \(\rightsarrow\) flexible notation
- Structure Editor for Terms \(\rightsarrow\) no ambiguities
The Nuprl System

Proof & program refinement in Type Theory

- Interactive Proof Editor \(\leadsto\) readable proofs
- Proof Tactics \(\leadsto\) user-defined inferences
- Decision Procedures \(\leadsto\) proof automation
- Flexible definition mechanism \(\leadsto\) user-defined terms
- Customizable Term Display \(\leadsto\) flexible notation
- Structure Editor for Terms \(\leadsto\) no ambiguities
- Library mechanism \(\leadsto\) user-theories
  - Large mathematical libraries and tactic collections
Proof & program refinement in Type Theory

- Interactive Proof Editor
  - readable proofs
- Proof Tactics
  - user-defined inferences
- Decision Procedures
  - proof automation
- Flexible definition mechanism
  - user-defined terms
- Customizable Term Display
  - flexible notation
- Structure Editor for Terms
  - no ambiguities
- Library mechanism
  - user-theories
  - Large mathematical libraries and tactic collections
- Program Extraction and Evaluation
  - program synthesis
A Platform for **Cooperating Reasoning Systems**

**Nuprl: System Architecture**

Basic System uses **Library**, **Editor**, and **Nuprl Refiner**
**Tactics: User-defined inference rules**

- **Meta-level programs built using**
  - Basic inference rules, standard tactics, predefined tacticals
  - Meta-level analysis of the proof goal and its context

\[\text{Applying a tactic always results in a valid proof}\]
Meta-level programs built using
- Basic inference rules, standard tactics, predefined tacticals
- Meta-level analysis of the proof goal and its context
→ Applying a tactic always results in a valid proof

Basic Tactics
\[ \text{Hypothesis: Prove } \ldots C \ldots \vdash C' \text{ where } C' \ \alpha\text{-equal to } C \]
- D \( c \): Decompose the outermost connective of clause \( c \)
- EqD \( c \): Decompose immediate subterms of an equality in clause \( c \)
- EqTypeD \( c \): Decompose type subterm of an equality in clause \( c \)
- Assert \( t \): Assert (or cut) term \( t \) as last hypothesis
- Auto: Apply trivial reasoning, decomposition, decision procedures
also rules tailored for: Logic, Induction, ...
Constructive Proofs and Program Extraction

Formal proof of Integer Square Root Theorem

\( \forall n \in \mathbb{N}. \ \exists r \in \mathbb{N}. \ r^2 \leq n < (r+1)^2 \)

BY allR

\( n \in \mathbb{N} \)
\( \vdash \exists r \in \mathbb{N}. \ r^2 \leq n < (r+1)^2 \)

BY NatInd 1

......basecase.....
\( \vdash \exists r \in \mathbb{N}. \ r^2 \leq 0 < (r+1)^2 \)
\( \checkmark \) BY existsR [0] THEN Auto

......upcase.....
\( i \in \mathbb{N}^+, \ r \in \mathbb{N}, \ r^2 \leq i-1 < (r+1)^2 \)
\( \vdash \exists r \in \mathbb{N}. \ r^2 \leq i < (r+1)^2 \)

BY Decide \( [(r+1)^2 \leq i] \) THEN Auto

......Case 1.....
\( i \in \mathbb{N}^+, \ r \in \mathbb{N}, \ r^2 \leq i-1 < (r+1)^2, \ (r+1)^2 \leq i \)
\( \vdash \exists r \in \mathbb{N}. \ r^2 \leq i < (r+1)^2 \)
\( \checkmark \) BY existsR [r+1] THEN Auto’

......Case 2.....
\( i \in \mathbb{N}^+, \ r \in \mathbb{N}, \ r^2 \leq i-1 < (r+1)^2, \ \neg((r+1)^2 \leq i) \)
\( \vdash \exists r \in \mathbb{N}. \ r^2 \leq i < (r+1)^2 \)
\( \checkmark \) BY existsR [r] THEN Auto
Algorithm Extracted from the Proof

• In raw Type Theory

\[
\text{let rec } \text{sqrt } i \\
= \text{if } i=0 \text{ then } <0, pf_i> \\
\text{else let } <r, pf_{i-1}> = \text{sqrt } (i-1) \\
\text{in} \\
\quad \text{if } (r+1)^2 \leq n \text{ then } <r+1, pf_i> \\
\text{else } <r, pf_i'>
\]

• In SML notation (after stripping proof components)

\[
\text{fun sqrt } n = \text{if } n=0 \text{ then } 0 \\
\quad \text{else let val } r = \text{sqrt } (n-1) \\
\quad \text{in} \\
\quad \quad \text{if } n<(r+1)^2 \text{ then } r \\
\quad \quad \text{else } r+1 \\
\quad \text{end}
\]
• Mathematically
  – Proof is short and “elegant” – why change it?
Are there better proofs?

- Mathematically
  - Proof is short and “elegant” – why change it?

- Computationally
  - Extracted algorithm for \( \lfloor \sqrt{n} \rfloor \) is linear in size of input \( n \) \( \mathcal{O}(n) \)
  - Proof uses standard induction on \( n \)
  - \( \forall P: \mathbb{N} \rightarrow \mathbb{P}. \ (P(0) \land (\forall i: \mathbb{N}^+. \ P(i-1) \Rightarrow P(i))) \Rightarrow (\forall i: \mathbb{N}. \ P(i)) \)
**Are there better proofs?**

- **Mathematically**
  - Proof is short and “elegant” – why change it?

- **Computationally**
  - Extracted algorithm for $\lceil \sqrt{n} \rceil$ is linear in size of input $n$ \( \mathcal{O}(n) \)
    Proof uses *standard induction* on $n$
    \[
    \forall P : N \rightarrow P. (P(0) \land (\forall i : N^+. P(i-1) \Rightarrow P(i))) \Rightarrow (\forall i : N. P(i))
    \]
  - A better algorithm would increase $r$ until $(r+1)^2 > n$ \( \mathcal{O}(\sqrt{n}) \)
    Corresponding proof needs schema for *bounded search*
    \[
    \forall P : N \rightarrow P. \forall n : N. P(n) \Rightarrow (\exists k : \{0..n\}. P(k) \land (\forall j : \{0..k-1\}. \neg P(j)))
    \]
Are there better proofs?

- **Mathematically**
  - Proof is short and “elegant” – why change it?

- **Computationally**
  - Extracted algorithm for $\lfloor \sqrt{n} \rfloor$ is linear in size of input $n \in \mathcal{O}(n)$
    - Proof uses *standard induction* on $n$
      \[
      \forall P: \mathbb{N} \rightarrow \mathbb{P}. \ (P(0) \land (\forall i: \mathbb{N}^+. \ P(i-1) \Rightarrow P(i))) \Rightarrow (\forall i: \mathbb{N}. \ P(i))
      \]
  - A better algorithm would increase $r$ until $(r+1)^2 > n \in \mathcal{O}(\sqrt{n})$
    - Corresponding proof needs schema for *bounded search*
      \[
      \forall P: \mathbb{N} \rightarrow \mathbb{P}. \ \forall n: \mathbb{N}. \ P(n) \Rightarrow (\exists k:\{0..n\}. \ P(k) \land (\forall j:\{0..k-1\}. \ \neg P(j)))
      \]
  - An even better algorithm computes $\lfloor \sqrt{n} \rfloor$ bit for bit $\in \mathcal{O}(\log_2 n)$
    - Proof almost identical to first one, but needs *4-adic induction*
      \[
      \forall P: \mathbb{N} \rightarrow \mathbb{P}. \ (P(0) \land (\forall i: \mathbb{N}. \ P(i \div 4) \Rightarrow P(i))) \Rightarrow (\forall i: \mathbb{N}. \ P(i))
      \]