[Part II]

Eli Barzilay


2

Scheme's Approach

- When it comes to implementing Scheme, these shortcuts are natural:
  - The implementation

The Problem

General Example

- Try to understand things following some very general example
- Assume some system S (any kind of system).
- In the systems we are interested in, the some

General Example

-

General Example

General

General Example

- Took some work off — no need for S_3', but still need to:
  - show that S_1' and S_2' are the same as those in S,
  - make sure

General Example: The Scheme Side

- This is exactly what happens in Scheme:
  - S_1: syntax structures,
  - S_2: evaluation function.
- Full scenario: a complete Scheme parser/interpreter in Scheme.
- Partial shortcut:
  - Create some new data structures in Scheme,
  - Make sure they are equivalent to Scheme's,
  - Write translation functions to (`unrep`) and from `rep` the

General Example: Conclusion

-
-
-

The Nuprl Side

- Moreover,

Term Quotations

- The suggested way of doing quotations was — add `quote` parameter

[redacted] Quotations: The Hard Way

- [redacted]lier: use a recursive type definition: an opid and a [redacted]
- Relies on having an available type theory, bad for Meta-PRL.
- [redacted]

Term Quotations: Another Alternative

- An example for some

Alternative: Problem #1

- First of all, we have to be really careful about the representation:

  in the above quotation there is no way to knoia.2795 d(t0d(v)Tj 9.hereed)T

Quoted Binding Variables

decision that binding variables are quoted with their

Terms with Alpha-Equality

- Using this frees us from making sure that no name-capture happens.
- For example, the Meta-PRL thinning rule looks like

$$\frac{H; J \vdash C}{H; x : A; J \vdash C}$$

  and this cannot be used when x appears in J (actual version is lines, compared to a previous ~ 20 line version).

Terms with Alpha-Equality

- A similar idea appears in Scheme macros.
- The usual problem is unsafe macros that capture names.
- For example:

```
(defmacro (swapf 1 55.5993 0 Txistapf 1 55.5679 0 Td((tmwapf 1 53
```

The End

- Related work: ?
- Conclusions: many.
- Future work: definitely!