Good and bad algebraic laws

Which right-hand sides are good?

\[(\log_{10} (10 \times m + d)) = (\log_{10} (10 \times m + d) + (\log_{10} (\frac{n}{10}) + 1)\]

\[(\log_{10} (10 \times m + d)) = (\log_{10} (10 \times m + d) + (\log_{10} (\frac{m}{10}) + 1)\]

\[(\log_{10} (10 \times m + d)) = (\log_{10} (10 \times m + d) + (\log_{10} m + 1)\]
Good and bad algebraic laws

Some parts are bad:

\[(\log_{10}(10 \times m + d)) \equiv (+ (\log_{10}(\frac{n}{10})) 1)\]

\[(\log_{10}(10 \times m + d)) \equiv (+ (\log_{10}(\frac{m}{10})) 1)\]

\[(\log_{10}(10 \times m + d)) \equiv (+ (\log_{10} m) 1)\]
Lists defined inductively

\( \text{LIST}(A) \) is the smallest set satisfying this equation:

\[
\text{LIST}(A) = \{ \text{'()} \} \cup \\{(\text{cons} \ a \ \text{as}) \mid a \in A, \text{as} \in \text{LIST}(A)\}
\]

Equivalently, \( \text{LIST}(A) \) is defined by these rules:

\[
\begin{align*}
\text{'()} & \in \text{List}(A) \\
\text{cons} \ a \ \text{as} & \in \text{List}(A)
\end{align*}
\]
One more inductive definition

A list of $A$ is one of:

• The empty list ’()’
• (cons $a$ as), where $a$ is an $A$ and as is a list of $A$
Lists generalized: S-expressions

An ordinary S-expression is one of:

- An atom (symbol, number, Boolean)
- A list of ordinary S-expressions

Can write literally in source, with quote
μScheme vs Impcore

New abstract syntax:
- LET (keyword, names, expressions, body)
- LAMBDA (formals, body)
- APPLY (exp, actuals)

New concrete syntax for LITERAL:
- (quote S-expression)
- ' S-expression
Introduce local names into environment

(let ([x1 e1]
      ...
      [xn en])
  e)

Square brackets mean the same as round, but are easier to see
What McCarthy might have done

(let ([val x1 e1]
  ...[
  [val xn en]])
  e)

(But semantics of let, let*, letrec is much simpler)