Meta

File: `C:\Users\Moses\math_ops\OperatorKernelO6\OperatorKernelO6\Meta\Meta.lean`
Type: lean
Generated: 2025-08-05 03:41:05
Size: 2279 characters

Overview

Meta-theoretical foundations

Source Code

```
import OperatorKernelO6.Kernel
import Mathlib.Data.Prod.Lex
import Mathlib.Tactic.Linarith

open OperatorKernelO6 Trace Step

namespace OperatorKernelO6.Meta

def deltaDepth : Trace → Nat
| void => 0
| delta t => deltaDepth t + 1
| integrate t => deltaDepth t
| merge a b => deltaDepth a + deltaDepth b
| recΔ _ _ n => deltaDepth n
| eqW a b => deltaDepth a + deltaDepth b

def recDepth : Trace → Nat
| void => 0
| delta t => recDepth t + 1
| integrate t => recDepth t
| merge a b => recDepth a + recDepth b
| recΔ b s n => deltaDepth n + recDepth b + recDepth s + 1
| eqW a b => recDepth a + recDepth b

def sz : Trace → Nat
| void => 1
| delta t => sz t + 1
| integrate t => sz t + 1
| merge a b => sz a + sz b + 1
| recΔ b s n => sz b + sz s + sz n + 1
| eqW a b => sz a + sz b + 1

def eqCount : Trace → Nat
| void => 0
| delta t => eqCount t
| integrate t => eqCount t
| merge a b => eqCount a + eqCount b
| recΔ b s n => eqCount b + eqCount s + eqCount n
| eqW a b => eqCount a + eqCount b + 1

def integCount : Trace → Nat
| void => 0
| delta t => integCount t
| integrate t => integCount t + 1
| merge a b => integCount a + integCount b
| recΔ b s n => integCount b + integCount s + integCount n
| eqW a b => integCount a + integCount b
```

```
def recCount : Trace → Nat
| void => 0
| delta t => recCount t
| integrate t => recCount t
| merge a b => recCount a + recCount b
| recΔ b s n => recCount b + recCount s + recCount n + 1
| eqW a b => recCount a + recCount b

def measure (t : Trace) : Prod Nat Nat := (recDepth t, sz t)

def lex (a b : Trace) : Prop :=
  Prod.Lex (· < ·) (· < ·) (measure a) (measure b)

def hasStep : Trace → Bool
| integrate (delta _) => true
| merge void _ => true
| merge _ void => true
| merge _ _ => true
| recΔ _ _ void => true
| recΔ _ _ (delta _) => true
| eqW _ _ => true
| _ => false

def tsize : Trace → Trace
| void => void
| delta t => delta (tsize t)
| integrate t => delta (tsize t)
| merge a b => delta (merge (tsize a) (tsize b))
| recΔ b s n => delta (merge (merge (tsize b) (tsize s)) (tsize n))
| eqW a b => delta (merge (tsize a) (tsize b))

def num0 : Trace := void
def num1 : Trace := delta void
def num2 : Trace := delta num1
def num3 : Trace := delta num2

def succ (t : Trace) : Trace := delta t

end OperatorKernelO6.Meta
```