THEORY
SUMandPRODUCT
TYPE PARAMETERS
T

AXIOMATIC DEFINITIONS
xdb1
OPERATORS
•SUM: SUM(s : T↔ℤ) EXPRESSION PREFIX
well-definedness condition
s ∈ T⇸ℤ

finite(s)

•PRODUCT: PRODUCT(s : T↔ℤ) EXPRESSION PREFIX
well-definedness condition
s ∈ T⇸ℤ

finite(s)

AXIOMS

axm1: SUM({p·p∈(T×ℤ)∧⊥|p}) = 0 // SUM(∅ ⦂ T↔ℤ) = 0

axm2: ∀t,x· t∈T ∧ x∈ℤ ⇒ SUM({t↦x}) = x

axm3: ∀s,t· s∈T⇸ℤ ∧ t∈T⇸ℤ ∧ s∩t=∅ ⇒ SUM(s∪t) = SUM(s) + SUM(t)

axm4: ⊤

axm5: PRODUCT({p·p∈(T×ℤ)∧⊥|p}) = 1

axm6: ∀t,x· t∈T ∧ x∈ℤ ⇒ PRODUCT({t↦x}) = x

axm7: ∀s,t· s∈T⇸ℤ ∧ t∈T⇸ℤ ∧ s∩t=∅ ⇒ PRODUCT(s∪t) = PRODUCT(s) ∗ PRODUCT(t)

THEOREMS

thm1 : ∀s,t· s∈T⇸ℤ ∧ t∈T⇸ℤ ∧ t⊆s ∧ finite(s) ⇒ SUM(s\t) = SUM(s) − SUM(t)

thm2 : ∀s,t· s∈T⇸ℤ ∧ t∈T⇸ℤ ∧ t⊆s ∧ finite(s) ∧ PRODUCT(t)≠0 ⇒ PRODUCT(s\t) = PRODUCT(s) ÷ PRODUCT

PROOF RULES
rulesBlock1 :
Metavariables
▪ s ∈ T↔ℤ

▪ t ∈ T↔ℤ

▪ x ∈ T

▪ y ∈ ℤ

Rewrite Rules
•rew1 : SUM(∅ ⦂ ℙ(T×ℤ)) (case-complete, interactive)
▪ rhs1 : ⊤ ► 0

•rew2 : SUM(s ∪ t) (case-incomplete, interactive)
▪ rhs1 : s∩t=∅ ► SUM(s) + SUM(t)

•rew3 : SUM({x↦y}) (case-complete, interactive)

▪ rhs1 : ⊤ ► y

•rew4 : SUM({x}⩤s) (case-incomplete, interactive)
▪ rhs1 : s∈T⇸ℤ ∧ x∈dom(s) ► SUM(s) − s(x)

rulesBlock2 :
Metavariables
▪ s ∈ T↔ℤ

▪ t ∈ T↔ℤ

▪ x ∈ T

▪ y ∈ ℤ

Rewrite Rules
•rew11 : PRODUCT(∅ ⦂ ℙ(T×ℤ)) (case-complete, interactive)
▪ rhs1 : ⊤ ► 1

•rew12 : PRODUCT(s ∪ t) (case-incomplete, interactive)
▪ rhs1 : s∩t=∅ ► PRODUCT(s) ∗ PRODUCT(t)

•rew13 : PRODUCT({x↦y}) (case-complete, interactive)
▪ rhs1 : ⊤ ► y

•rew14 : PRODUCT({x}⩤s) (case-incomplete, interactive)
▪ rhs1 : s∈T⇸ℤ ∧ x∈dom(s) ∧ s(x)≠0 ► PRODUCT(s) ÷ s(x)

END

```
MACHINE
        m1      ›
SEES
        c1
VARIABLES
        values    ›
INVARIANTS
        inv1:    values ∈ DATA ⇸ ℤ not theorem ›
        inv2:    finite(values) not theorem ›
EVENTS
        INITIALISATION:   not extended ordinary ›
                THEN
                        act1:    values ≔ ∅ ›
                END

        AddValue:         not extended ordinary ›
                ANY
                        d        ›
                        v        ›
                WHERE
                        grd1:    d ∉ dom(values) not theorem ›
                        grd2:    v ∈ ℤ not theorem ›
                THEN
                        act1:    values ≔ values ∪ {d↦v} ›
                END

        RemoveValue:      not extended ordinary ›
                ANY
                        v        ›
                WHERE
                        grd1:    v ∈ dom(values) not theorem 〉
                THEN
                        act1:    values ≔ {v}⩤values ›
                END

        GetSum:   not extended ordinary ›
                ANY
                        result   ›
                WHERE
                        grd1:    result = SUM(values) not theorem ›result = sum(values)
                END

        GetProduct:       not extended ordinary ›
                ANY
                        result   ›
                WHERE
                        grd1:    result = PRODUCT(values) not theorem ›
                END

END
```

```
MACHINE
        m2      ›
REFINES
        m1
SEES
        c1
VARIABLES
        values  ›
        s       ›
        p       ›
INVARIANTS
        inv1:   s = SUM(values) not theorem ›
        inv2:   p = PRODUCT(values) not theorem ›
        inv3:   0 ∉ ran(values) not theorem ›
EVENTS
        INITIALISATION:   extended ordinary ›
                THEN
                        act1:   values ≔ ∅ ›
                        act2:   s ≔ 0 ›
                        act3:   p ≔ 1 ›
                END

        AddValue:         extended ordinary ›
                REFINES
                          AddValue
                ANY
                        d       ›
                        v       ›
                WHERE
                        grd1:   d ∉ dom(values) not theorem ›
                        grd2:   v ∈ ℤ not theorem ›
                        grd3:   v ≠ 0 not theorem ›
                THEN
                        act1:   values ≔ values ∪ {d↦v} ›
                        act2:   s ≔ s + v ›
                        act3:   p ≔ p ∗ v ›
                END

        RemoveValue:      extended ordinary ›
                REFINES
                          RemoveValue
                ANY
                        v       ›
                WHERE
                        grd1:   v ∈ dom(values) not theorem ›
                THEN
                        act1:   values ≔ {v}⩤values ›
                        act2:   s ≔ s − values(v) ›
                        act3:   p ≔ p ÷ values(v) ›
                END

        GetSum:   not extended ordinary ›
                REFINES
                          GetSum
                ANY
                        result  ›
                WHERE
                        grd1:   result = s not theorem ›
                END

        GetProduct:       not extended ordinary ›
                REFINES
                          GetProduct
                ANY
                        result  ›
```

```
WHERE
        grd1:     result = p not theorem ›
END

END
```