Physical units part II

Reminder : physical units

A unit expression is defined by following grammar

$u, v := b \mid 1 \mid u * v \mid u^{-1}$

where u, v are unit expressions themselves and b is a base unit:

$b := m \mid kg \mid s \mid A \mid K \mid cd \mid mol$

You may use B to denote the set of the unit types

$B = \{ m, kg, s, A, K, cd, mol \}$

For readability reasons, we use the syntactic sugar

$u\hat{\ }n = u * \ldots * u$ if $n > 0$

1 if $n = 0$

$u^{-1} * \ldots * u^{-1}$ if $n < 0$

and $u/v = u * v^{-1}$

Theorem

Theorem:

Suppose that the result type T does not contain a base unit b_1 (or, equivalently, this base type occurs only with the overall exponent 0, as b_1^0). If we multiply all variables of type b_1 by a fixed numerical constant K, the final result of the expression does not change.

Question 1: Generalize the theorem.

Question 2: Prove the theorem.

Theorem examples

- Center of mass

x: <m> = (x1*m1 + x2*m2)/(m1+m2)

Changing all mass variables by 2 does not change the center of mass.

- Gravity estimation

$t_1 = \surd(t_2 \hat{} 2 * h_1/h_2)$

Changing h_1 and h_2 by any factor will not change the time.

Reminder: Rules

$$\frac{\Gamma \vdash a : U}{\Gamma \vdash a : \mathrm{simplify}(U)}$$

$$\frac{\Gamma \vdash a : (U*U)^{-1}}{\Gamma \vdash \sqrt{a} : U^{-1}}$$

$$\frac{\Gamma \vdash a : U \qquad \Gamma \vdash b : U}{\Gamma \vdash a + b : U}$$

$$\frac{\Gamma \vdash a : U \qquad \Gamma \vdash b : V}{\Gamma \vdash a * b : U*V}$$

$$\frac{\Gamma \vdash a : U \qquad \Gamma \vdash b : V}{\Gamma \vdash a / b : U/V}$$

$$\frac{\Gamma \vdash a : U*U}{\Gamma \vdash \sqrt{a} : U}$$

$$\frac{\Gamma \vdash a : 1}{\Gamma \vdash \sin(a) : 1}$$

$$\frac{\Gamma \vdash a : U}{\Gamma \vdash \mathrm{abs}(a) : U}$$

Solution – part I

Lemma:

If we multiply all variables of type B by constant K, and the result has type T where B has exponent N, then the value of the expression is multiplied by K^N

Solution – part II

Suppose that we have proved the lemma for expressions of size $< n$.

Let give us an expression of size n. If the last applied rule is +, then:

$$\frac{\Gamma \vdash E : U \qquad \Gamma \vdash F : U}{\Gamma \vdash E + F : U}$$

Let us assume that B appears in U with exponent N.

If we multiply all variables of type B in E and F by K, by recurrence E is multiplied by K^n.

Therefore E+F is transformed to (E*K^N + F*K^N) = (E+F)*K^N

Other rules are similar.

Type inference for physical units

```
val g = 9.87.m / (1.s * 1.s)
val href = 1.92.m

def f(x, y, z) = sqrt(x/y) + z
def T(L) = 2*pi*sqrt(L/g) + 0.s
def fall(t, v) = -0.5*g*t*t + t*v + href
def freq(t, w) =
   href * sin(2*pi*t*w) + g*(t/w)
def ein(E, p, v) = E - p*v*v == 0 &&
   E/1.s - p*g*v == 0 && p > 0.kg
def prof(a, b) = if(b > 1.s)
   sqrt(prof(a+a,b-1.s)/b) else a
```

```
def f(x, y, z) = sqrt(x/y) + z
      TX TY TZ          TX TY
              :TR        Tx/y
                 Tsqrt
                       T+
```

T+ = TR

Tsqrt = T+

TZ = T+

Tsqrt * Tsqrt = Tx/y

Tx/y = TX / TY

Therefore:

Tsqrt = TR

TZ = TR

TR*TR = TX / TY

so **TX = TR*TR/TY**

def f[T,V](x: T*T/V, y:V, z: T): T

```
def T(L)     = 2pi*sqrt(L/g) + 0.s
      TL :TR    1       TLG
                     TSQRT
                  TM
                             TP
```

TR = TP

TP = s

TP = TM

TM = 1 * TSQRT

TSQRT*TSQRT = TLG

TLG = TL / (m/(s*s))

Therefore:

TR = s

s = TQSRT

s * s = TLG

s*s = TL / (m/(s*s))

So: **TL = m** and **T(L: <m>): <s>**

```
def fall(t, v)      = (-0.5*g*t*t + t*v) + href
         TT TV :TR        1
                          TM1          TM4
                            TM2
                               TM3
                                TP1
                                     TP2
```

```
TR = TP2, TM1 = 1*m/(s*s), TM2 = TM1*TT, TM3 = TM2*TT,
TM4 = TT*TV, TP1 = TM4, TP1 = TM3, TP2 = TP1, TP2 = m
```

```
Therefore:
```

```
TM2 = m/(s*s)*TT, TM3 = m/(s*s)*TT*TT, TP1 = TT*TV,
TP1 = m/(s*s)*TT*TT, TT*TV = m/(s*s)*TT*TT
```

```
TV = m/(s*s)*TT
```

TR = m, TP1 = m, so m = m/(s*s)*TT*TT, therefore **TT=s**

TV = m/s

fall(t: <s>): <m>

```
def freq(t, w)     = href * sin(2pi*t*w)    + g*(t/w)
         TT TW: TR     m     1    1  TT TW  m/s^2 TTOW
                                   TTW         TM2
                           TM1
                                      TP
```

TP = TM1, TM1 = TM2, TM2 = m/s^2*TTOW, TTOW=TT/TW, TTW = TT*TW, TTW = 1, TM1 = m*1, TR = TP

Therefore

TR = m

m = m/s^2 * TTOW, so TT/TW = s^2

1 = TT * TW

so TW = 1/TT and TT^2 = s^2, so **TT = s and TW = 1/s**

def freq(t: <s>, w: <1/s>): <m>

```
def ein(E, p, v): Bool =
        TE TP TV
E - p*v*v == 0 && E/1.s - p*g*v == 0 && p > 0.kg
TE   TM1              TES     TM2
```

TP = kg, TE = TM1, TM1 = TP * TV^2, TES = TE/s, TES = TM2, TM2 = TP*m/s^2*TV

TE/s = TM2, TE/s = kg*m/s^2*TV, TE = kg*TV^2

So kg*TV^2/s = kg*m/s^2*TV

therefore: **TV = m/s, TE = kg*m^2/s^2**

def ein(E: <kg*m^2/s^2>, p: <kg>, v: <m/s>): Bool

```
def prof(a, b) =
         TA TB   :TR
if(b > 1.s) sqrt(prof(a+a,b-1.s)/b) else a
    BOOL              TPA TB1
                      TR
                          TRB
                   TSQRT
```

TB = s

TSQRT = TA = TR

TQSRT*TSQRT = TRB

TRB = TR / TB

TA = TA = TA = TA, TB = s = TB

TR * TR = TR / s

so **TR = 1/s, TA = 1/s**

def prof(a: <1/s>, b: <s>): <1/s>

Type inference

Bonus:

```
Type check omega function
def w(f)(x) = f(w(f)(f(x)))

def w(f:TF)(x:TX):TR = f(w(f)(f(x):TA):TB)


TF = TX => TA
TA = TX
omega: TF x TX => TR
TR = TB
TF = TB => TR
Therefore : TF = TX => TX
TX => TX == TB => TB
TB = TX
def w[T](f: T => T, x: T): T
```