Exercise

November 19, 2014

Units as Types | 1

A *unit expression* is defined by the following grammar

$$\begin{aligned} unit &:= \quad B \quad | \quad 1 \quad | \quad unit * unit \quad | \quad unit^{-1} \\ B &:= \quad m \quad | \quad kg \quad | \quad s \quad | \quad A \quad | \quad K \quad | \quad cd \quad | \quad mole \end{aligned}$$

where, *cd* denotes *candela* - unit of luminous intensity, A - Ampere, K - kelvin and *mol* denotes *mole* - measure of amount of atoms, molecules etc.
For readability, we use the syntactic sugar

$$u^n = \begin{cases} u * ... * u & n > 0 \\ 1 & n = 0 \\ u^{-1} * ... * u^{-1} & n < 0 \end{cases} \tag{1}$$

and

$$\frac{u}{v} = u * v^{-1}$$

a) Give the type rules for the arithmetic operations $+$, $*$, $/$, $\sqrt{\ }$, sin , *abs*, that satisfy the following conditions: (a) Only expressions having the same units can be added. The result will have the same unit as the operands. (b) Multiplying two expressions multiplies their units. Similarly, for division. (c) Square root can be applied only over expressions whose units are of the form u^2, for some unit u. (d) trigonometric functions take as argument radians, which are dimensionless (since they are defined as the ratio of arc length to radius). You can denote that a variable is dimensionless by $\Gamma \vdash e : 1$.

b) The unit expressions as defined above are strings, so e.g.

$$\frac{s^4 m^2}{s^2 m^3} \neq \frac{s^2}{m}$$

however physically these units match. Define a procedure such that your type rules type check expressions, whenever they are correct according to physics.

c) Determine the type of T in the following code fragment. The values in angle brackets give the unit type expressions of the variables and Pi is the usual constant π in the Scala math library. Give the full type derivation tree using your rules from a) and b), i.e. the tree that infers the types of the variables R, w, T.

```
val x: <m> = 800
val y: <m> = 6378
val g: <m/(s*s)> = 9.8
val R = x + y
val w = sqrt(g/R)
val T = (2 * Pi) / w
```

d) Consider the following function that computes the Coulomb force, and suppose for now that the compiler can parse the type expressions:

```
def coulomb(k: <(N*m)/(C*C)>, q1: <C>, q2: <C>, r: <m>): <N> {

  return (k* q1 * q2)/(r*r)

}
```

The derived types are defined as $C = As$ and $N = \frac{kg\ m}{s^2}$. Does the code type check? Justify your answer.

e) Suppose you want to use the unit *feet* in addition to the Si units above. How can you extend your type system to accommodate for this? Explain your answer.
(Assume that 1m = 3.28084 feet.)

Make sure that your extension makes physical sense. For example, if $x : feet$ and $y : m$, then adding them directly as $x + y$ is illegal. On the other hand, it should still be possible to add together quantities that can be meaningfully added together (do not make your system more restrictive than necessary).