States in FRACTRAN

By Aaron Berger for HSCTF 2014

FRACTRAN: A programming language that uses a list of fractions as code and takes input n. To execute, for the first fraction *f* in the list for which *nf* is an integer, replace *n* by *nf*, repeat this rule until no fraction in the list produces an integer when multiplied by *n*, then halt (Wikipedia).

Let's say you need to write a FRACTRAN program which copies a number **a**. Our program will start with input 2^a and return $2^a \times 3^a$. Your first guess might to be to write the program $\left\{\frac{2\times3}{2}\right\}$. Won't this divide by 2 and then multiply by 2 and 3, thus converting all the 2s into equal amounts of 2s and 3s? Nope. Let's try running it.

Program: $\left\{\frac{6}{2}\right\}$

Input: $8=2^3$

Step 1: $8 \times \frac{6}{2} = 24 = 2^3 \times 3^1$.

Step 2: $24 \times \frac{6}{2} = 72 = 2^3 \times 3^2$

Step 3: $72 \times \frac{6}{2} = 216 = 2^3 \times 3^3$

And so on.

First of all, we can see that $\frac{6}{2}$ reduces to $\frac{3}{1}$, which we can always multiply by and still get an integer, so clearly the program will never terminate. However, the underlying problem is that the power of 2 is never decreasing. $8=(2^3)$. $24=(2^3)\times(3)$. $72=(2^3)\times(3^2)$. $216=(2^3)\times(3^3)$. If we never get rid of the 2s, our program will run indefinitely. To fix this, we need change our method slightly. Here's the general idea:

1. Convert all 2s to 3s and 5s. (Using the fraction $\frac{3\times5}{2}$).

2. Convert all 5s back to 2s (Using the fraction $\frac{2}{5}$).

Now, we can't just put these two fractions together and make a working program. Let's see why not:

Program: $\left\{\frac{15}{2}\right\}, \left\{\frac{2}{5}\right\}$

Input: $4 = 2^2$

Step 1: $4 \times \frac{15}{2} = 30 = 2^1 \times 3^1 \times 5^1$

Step 2: $30 \times \frac{15}{2} = 225 = 3^2 \times 5^2$

All good so far. Now to convert the 5s back to 2s:

Step 3: $225 \times \frac{2}{5} = 90 = 2^1 \times 3^2 \times 5^1$

Step 4: $90 \times \frac{15}{2} = 675 = 3^3 \times 5^2$

Not good! Instead of finishing the conversion of the 5s back to 2s, the program converted the newly-made 2 right back into a 3 and a 5. It's easy to see that from here the program will get stuck in a loop converting a 2 into a 3 and a 5 and then converting the 5 back into a 2. We're closer to the solution now, but we still need to stop the program from converting these newly-made 2s into 3s and 5s.

To fix this, we will use states. States work just like LABEL and GOTO commands, as shown below:

1. State A: Convert all 2s to 3s and 5s. (Using the fraction $\frac{3\times5}{2}$), **GOTO A**

2. State A: If you can't multiply by $\frac{3\times5}{2}$, **GOTO B**

3. State B: Convert all 5s back to 2s (Using the fraction $\frac{2}{5}$), **GOTO B**

4. State B: If you can't multiply by $\frac{2}{5}$, **terminate**.

To create a state, you simply need a pair of state markers. We'll use **7** and **11** for state **A**, and **13** and **17** for state **B**. The two markers alternate between being in the numerator and denominator, so that you can't perform the action(s) associated with that state unless your number is divisible by one of the two markers. The instructions would be written as follows:

1. $\frac{3\times5\times\mathbf{7}}{2\times\mathbf{11}}, \frac{\mathbf{11}}{\mathbf{7}}$

2. $\frac{\mathbf{17}}{\mathbf{11}}$

3. $\frac{2\times\mathbf{13}}{5\times\mathbf{17}}, \frac{\mathbf{17}}{\mathbf{13}}$

4. $\frac{1}{\mathbf{17}}$

The compiled program looks like this: $\frac{3\times5\times\mathbf{7}}{2\times\mathbf{11}}, \frac{\mathbf{11}}{\mathbf{7}}, \frac{\mathbf{17}}{\mathbf{11}}, \frac{2\times\mathbf{13}}{5\times\mathbf{17}}, \frac{\mathbf{17}}{\mathbf{13}}, \frac{1}{\mathbf{17}}$

As a final comment before we put this to the test, note that there is no way for this program to get to state **A**. That's fine; the user will just have to do this manually by appending either a 7 or an 11 to the input (in fact, according to the specifications of the problem, there is no way to write it so that the user won't have to append a state marker to the input—can you prove why?). Let's put it to the test:

Program: $\left\{\frac{105}{22}\right\}, \left\{\frac{11}{7}\right\}, \left\{\frac{17}{11}\right\}, \left\{\frac{26}{85}\right\}, \left\{\frac{17}{13}\right\}, \left\{\frac{1}{17}\right\}$

Input: 44 = $2^2\times\mathbf{11}^1$

Step 1: $44\times\frac{105}{22} = 210 = 2^1\times3^1\times5^1\times\mathbf{7}^1$

Step 2: $210\times\frac{11}{7} = 330 = 2^1\times3^1\times5^1\times\mathbf{11}^1$

Step 3: $330\times\frac{105}{22} = 1575 = 2^0\times3^2\times5^2\times\mathbf{7}^1$

Step 4: $1575\times\frac{11}{7} = 2475 = 2^0\times3^2\times5^2\times\mathbf{11}^1$

Everything is looking good so far. The 2s have been converted into 3s and 5s and the state markers, 7 and 11, are alternating properly.

Step 5: $2475\times\frac{17}{11} = 3825 = 2^0\times3^2\times5^2\times\mathbf{17}^1$

Step 6: $3825\times\frac{26}{85} = 1170 = 2^1\times3^2\times5^1\times\mathbf{13}^1$

Step 7: $1170\times\frac{17}{13} = 1530 = 2^1\times3^2\times5^1\times\mathbf{17}^1$

Step 8: $1530\times\frac{26}{85} = 468 = 2^2\times3^2\times5^0\times\mathbf{13}^1$

Step 9: $468\times\frac{17}{13} = 612 = 2^2\times3^2\times5^0\times\mathbf{17}^1$

Step 10: $612\times\frac{1}{17} = 36 = 2^2\times3^2$

And we have successfully copied the input from the 2s to the 3s with the help of states.