FRACTRAN: A SIMPLE UNIVERSAL PROGRAMMING LANGUAGE FOR ARITHMETIC

J.H. Conway

Department of Mathematics
Princeton University
Princeton, NJ 08544

1. Your Free Samples of FRACTRAN.

To play the *fraction game* corresponding to a given list

$$f_1, f_2, \ldots, f_k$$

of fractions and starting integer N, you repeatedly multiply the integer you have at any stage (initially N) by the earliest f_i in the list for which the answer is integral. Whenever there is no such f_i, the game *stops*.

(Formally, we define the sequence $\{N_n\}$ by $N_0 = N$, $N_{n+1} = f_i N_n$, where i $(1 \le i \le k)$ is the least i for which $f_i N_n$ is integral, as long as such an i exists.)

Theorem 1: When PRIMEGAME:

$$\frac{17}{91}\ \frac{78}{85}\ \frac{19}{51}\ \frac{23}{38}\ \frac{29}{33}\ \frac{77}{29}\ \frac{95}{23}\ \frac{77}{19}\ \frac{1}{17}\ \frac{11}{13}\ \frac{13}{11}\ \frac{15}{2}\ \frac{1}{7}\ \frac{55}{1}$$

is started at 2, the other powers of 2 that appear, namely,

$$2^2, 2^3, 2^5, 2^7, 2^{11}, 2^{13}, 2^{17}, 2^{19}, 2^{23}, 2^{29}, \ldots,$$

are precisely those whose indices are the prime numbers, in order of magnitude.

Theorem 2: When PIGAME:

$$\frac{365}{46}\ \frac{29}{161}\ \frac{79}{575}\ \frac{679}{451}\ \frac{3159}{413}\ \frac{83}{407}\ \frac{473}{371}\ \frac{638}{355}\ \frac{434}{335}\ \frac{89}{235}\ \frac{17}{209}\ \frac{79}{122}$$

$$\frac{31}{183}\ \frac{41}{115}\ \frac{517}{89}\ \frac{111}{83}\ \frac{305}{79}\ \frac{23}{73}\ \frac{73}{71}\ \frac{61}{67}\ \frac{37}{61}\ \frac{19}{59}\ \frac{89}{57}\ \frac{41}{53}\ \frac{833}{47}\ \frac{53}{43}$$

$$\frac{86}{41}\ \frac{13}{38}\ \frac{23}{37}\ \frac{67}{31}\ \frac{71}{29}\ \frac{83}{19}\ \frac{475}{17}\ \frac{59}{13}\ \frac{41}{291}\ \frac{1}{7}\ \frac{1}{11}\ \frac{1}{1024}\ \frac{1}{97}\ \frac{89}{1}$$

is started at 2^n, the next power of 2 to appear is $2^{\pi(n)}$, where for

n =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20 .
$\pi(n)$ =	3	1	4	1	5	9	2	6	5	3	5	8	9	7	9	3	2	3	8	4	6 .

For an arbitrary natural number n, $\pi(n)$ is the nth digit after the point in the decimal expansion of the number π.

Theorem 3: Define $f_c(n) = m$ if POLYGAME:

$$\frac{583}{559}\ \frac{629}{551}\ \frac{437}{527}\ \frac{82}{517}\ \frac{615}{329}\ \frac{371}{129}\ \frac{1}{115}\ \frac{53}{86}\ \frac{43}{53}\ \frac{23}{47}\ \frac{341}{46}$$

$$\frac{41}{43}\ \frac{47}{41}\ \frac{29}{37}\ \frac{37}{31}\ \frac{37}{31}\ \frac{299}{29}\ \frac{47}{23}\ \frac{161}{15}\ \frac{527}{19}\ \frac{159}{7}\ \frac{1}{17}\ \frac{1}{13}\ \frac{1}{3}$$

when started at $c2^{2^n}$, stops at 2^{2^m}, and otherwise leave $f_c(n)$ undefined. Then every computable function appears among $f_0, f_1, f_2, \ldots$.

2. The Catalogue.

We remark that the "catalogue numbers" c are easily computed for some quite interesting functions. Table 1 and its notes give f_c for any c whose largest odd divisor is less than $2^{10} = 1024$.

Table 1. The Catalogue

c	All defined values of f_c
0	none
1	$n \to n$
2	$0 \to 1$
4	$0 \to 2$
8	$1 \to 2$
16	$2 \to 3$
64	$1 \to 3$
77	$n \to 0$
128	$0 \to 3$
133	$0 \to 0$
255	$n+1 \to n+1$
256	$3 \to 4$
847	$n \to 1$
37485	$0 \to 0,\ n+1 \to n$
2268945	$n \to n+1$
2^k	$a \to b$ if $2^b - 2^a = k$
$7 \cdot 11^{2^k}$	$n \to k$
$\frac{15}{7} \cdot 1029^{2^{k-1}}$	$n \to n+k$
c_π	$n \to \pi(n)$

In this Table, n denotes an arbitrary non-negative integer.

We also have

$$f_{2^k A} = f_0 \,;$$

$$f_{2^k B} = f_{2^k} \,;\ f_{2^k B'} = f_{2^{k+1}} \,;$$

$$f_{2^k C} = f_{77} \,;\ f_{2^k C'} = f_{847} \,;$$

$$f_{2^k D} = f_{133} \ \ (k=0) \ \text{ or } \ f_0 \ \ (k>0) \,;$$

$$f_{2^k E} = f_{255} \ \ (k=0) \ \text{ or } \ f_{2^k} \ \ (k>0) \,;$$

where

A	is	any odd number < 1024 not visible below:
B	is	1,3,9,13,17,27,39,45,51,81,105,115,117,135,145,153,155, 161,169,185,195,203,205,217,221,235,243,259,287,289,315, 329,345,351,405,435,459,465,483,507,555,585,609,615,651, 663,705,729,777,861,945,975,987,1017, . . .
B'	is	165,495, . . .
C	is	77,91,231,273,385,455,539,1015, . . .
C'	is	847, 1001, . . .
D	is	133, 285, 399, 665, 855, . . .
E	is	255, ...

Figure 1 gives a c for which $f_c(n)$ is the above function $\pi(n)$

$$3^{\begin{array}{l}
2^{100!} + 2^{\frac{365}{46}101\cdot 100!} + 2^{\frac{29}{161}101^{2}100!} + 2^{\frac{79}{575}101^{3}100!} + 2^{\frac{7}{451}101^{4}100!} \\
+ 2^{\frac{3159}{413}101^{5}100!} + 2^{\frac{83}{407}101^{6}100!} + 2^{\frac{473}{371}101^{7}100!} + 2^{\frac{638}{355}101^{8}100!} + 2^{\frac{434}{335}101^{9}100!} \\
+ 2^{\frac{89}{235}101^{10}100!} + 2^{\frac{17}{209}101^{11}100!} + 2^{\frac{79}{122}101^{12}100!} + 2^{\frac{31}{183}101^{13}100!} + 2^{\frac{41}{115}101^{14}100!} \\
+ 2^{\frac{517}{89}101^{15}100!} + 2^{\frac{111}{83}101^{16}100!} + 2^{\frac{305}{79}101^{17}100!} + 2^{\frac{23}{73}101^{18}100!} + 2^{\frac{73}{71}101^{19}100!} \\
+ 2^{\frac{61}{67}101^{20}100!} + 2^{\frac{37}{61}101^{21}100!} + 2^{\frac{19}{59}101^{22}100!} + 2^{\frac{89}{57}101^{23}100!} + 2^{\frac{41}{53}101^{24}100!} \\
+ 2^{\frac{833}{47}101^{25}100!} + 2^{\frac{53}{43}101^{26}100!} + 2^{\frac{86}{41}101^{27}100!} + 2^{\frac{13}{38}101^{28}100!} + 2^{\frac{23}{37}101^{29}100!} \\
+ 2^{\frac{67}{31}101^{30}100!} + 2^{\frac{71}{29}101^{31}100!} + 2^{\frac{83}{19}101^{32}100!} + 2^{\frac{475}{17}101^{33}100!} + 2^{\frac{59}{13}101^{34}100!} \\
+ 2^{\frac{41}{3}101^{35}100!} + 2^{\frac{1}{7}101^{36}100!} + 2^{\frac{1}{11}101^{37}100!} + 2^{\frac{1}{1024}101^{38}100!} + 2^{101^{39}100!}
\end{array}} \times 5^{2^{89\cdot 101!}+2^{90\cdot 101!}} \times 17^{101!-1} \times 23$$

Figure 1. The constant c_π .

3. Avoid Brand X.

Works that develop the theory of effective computation are often written by authors whose interests are more logical than computational, and so they seldom give elegant treatments of the essentially computational parts of this theory. Any effective enumeration of the computable functions is probably complicated enough to spread over a chapter, and we might read that "of course the explicit computation of the index number for any function of interest is totally impracticable." Many of these defects stem from a bad choice of the underlying computational model.

Here we take the view that it is precisely because the particular computational model has no great logical interest that it should be carefully chosen. The logical points will be all the more clear when they don't have to be disentangled by the reader from a clumsy program written in an awkward language, and we can then "sell" the theory to a wider audience by giving simple and striking examples explicitly. (It is for associated reasons that we use the easily comprehended term "computable function" as a synonym for the usual "partial recursive function.")

4. Only FRACTRAN Has These Star Qualities.

FRACTRAN is a simple theoretical programming language for arithmetic that has none of the defects described above.

- *Makes workday really easy!*

FRACTRAN needs no complicated programming manual - its entire syntax can be learned in 10 seconds, and programs for quite complicated and interesting functions can be written almost at once.

- *Gets those functions really clean!*

The entire configuration of a FRACTRAN machine at any instant is held as a single integer - there are no messy "tapes" or other foreign concepts to be understood by the fledgling programmer.

- *Matches any machine on the market!*

Your old machines (Turing, etc.) can quite easily be made to simulate arbitrary FRACTRAN programs, and it is usually even easier to write a FRACTRAN program to simulate other machines.

- *Astoundingly simple universal program!*

By making a FRACTRAN program that simulates an arbitrary other FRACTRAN program, we have obtained the simple universal FRACTRAN program described in Theorem 3.

5. Your PRIMEGAME Guarantee!

In some ways, it is a pity to remove some of the mystery from our programs such as PRIMEGAME. However, it is well said [2] that "A mathematician is a conjurer who gives away his secrets," so we'll now prove Theorem 1.

To help in Figure 2, we have labeled the fractions:

A	B	C	D	E	F	G	H	I	J	K	L	M	N
$\frac{17}{91}$	$\frac{78}{85}$	$\frac{19}{51}$	$\frac{23}{38}$	$\frac{29}{33}$	$\frac{77}{29}$	$\frac{95}{23}$	$\frac{77}{19}$	$\frac{1}{17}$	$\frac{11}{13}$	$\frac{13}{11}$	$\frac{15}{2}$	$\frac{1}{7}$	$\frac{55}{1}$

and we note that $AB = \frac{2 \times 3}{5 \times 7}$, $EF = \frac{7}{3}$, $DG = \frac{5}{2}$.

We let n and d be numbers with $0 < d < n$ and write $n = qd + r$ $(0 \le r < d)$. Figure 2 illustrates the action of PRIMEGAME on the number $5^n 7^d 13$. We see that this leads to $5^n 7^{d-1} 13$ or $5^{n+1} 7^n 13$ according as d does or does not divide n. Moreover, the only case when a power of 2 arises is as the number $2^n 7^{d-1}$ when $d = 1$.

$$
\begin{tikzcd}
& 5^n 7^d 13 \arrow[d, "(AB)^d J"] & \\
& 2^d 3^d 5^{n-d} 11 \arrow[d, "(EF)^d K"] & \\
& 2^d 5^{n-d} 7^d 13 \arrow[d, "(AB)^d J"] & \\
& 2^{2d} 3^d 5^{n-2d} 11 \arrow[d, "(EF)^d K"] & \\
& 2^{2d} 5^{n-2d} 7^d 13 \arrow[d, "(AB)^d J"] & \\
& \vdots \arrow[d, "(EF)^d K"] & \\
& 2^{qd} 5^r 7^d 13 \arrow[d, "(AB)^r A"] & \\
& 2^n 3^r 7^{d-r-1} 17 \arrow[dl, "r>0"', "C"] \arrow[dr, "r=0", "I"'] & \\
2^n 3^{r-1} 7^{d-r-1} 19 \arrow[d, "(DG)^n H"] & & 2^n 7^{d-1} \arrow[d, "L^n M^{d-1} N"] \\
3^{r-1} 5^n 7^{d-r} 11 \arrow[d, "(EF)^{r-1} K"] & & 3^n 5^{n+1} 11 \arrow[d, "(EF)^n K"] \\
5^n 7^{d-1} 13 & & 5^{n+1} 7^n 13
\end{tikzcd}
$$

Figure 2. The action of PRIMEGAME.

It follows that when the game is started at $5^n 7^{n-1} 13$, it tests all numbers from $n-1$ down to 1 until it first finds a divisor of n, and then continues with n increased by 1. In the process, it passes through a power of 2^n of 2 only when the largest divisor of n that is less than n is $d = 1$, or in other words, only when n is prime.

6. FRACTRAN - Your Free Introductory Offer.

A FRACTRAN program may have any number of lines, and a typical line might have the form

$$\text{line } 13: \frac{2}{3} \to 7, \ \frac{4}{5} \to 14 .$$

At this line, the machine replaces the current working integer N by $\frac{2}{3}N$, if this is again an integer, and goes to line 7. If $\frac{2}{3}N$ is not an integer, but $\frac{4}{5}N$ is, we should instead replace N by $\frac{4}{5}N$, and go to line 14. If neither $\frac{2}{3}N$ nor $\frac{4}{5}N$ is integral, we should *stop* at line 13.

More generally, a FRACTRAN program line has the form

$$\text{line } n: \frac{p_1}{q_1} \to n_1, \ \frac{p_2}{q_2} \to n_2, \ldots, \frac{p_k}{q_k} \to n_k .$$

The action of the machine at this line is to replace N by $\frac{p_i}{q_i}N$ for the least i $(1 \le i \le k)$ for which this is integral, and then go to line n_i ; or, if no $\frac{p_i}{q_i}N$ is integral, to *stop* at line n. (A line with $k = 0$ is permitted and serves as an unconditional stop order.)

A FRACTRAN program that has just n lines is called a FRACTRAN-n program. We introduce the convention that a line that cannot be jumped to counts as a $\frac{1}{2}$-line. (Sensible programs will contain at most one $\frac{1}{2}$-line, the initial line.)

We write

$$[\ \frac{p_1}{q_1}\ \frac{p_2}{q_2}\ \cdots\ \frac{p_k}{q_k}\ ]$$

for the FRACTRAN-1 program

$$\text{line } 1: \frac{p_1}{q_1} \to 1\,, \ \frac{p_2}{q_2} \to 1, \ldots, \frac{p_k}{q_k} \to 1\,.$$

We shall see that every FRACTRAN program can be simulated by a FRACTRAN-1 program which starts at a suitable multiple of the original starting number. With a FRACTRAN-$1\frac{1}{2}$ program, we can make this multiple be 1.

The FRACTRAN-$1\frac{1}{2}$ program

$$\text{line } 0: \frac{P_1}{Q_1} \to 1\,, \ \frac{P_2}{Q_2} \to 1, \ldots, \frac{P_j}{Q_j} \to 1$$

$$\text{line } 1: \frac{p_1}{q_1} \to 1\,, \ \frac{p_2}{q_2} \to 1, \ldots, \frac{p_k}{q_k} \to 1$$

is symbolized by

$$\frac{P_1}{Q_1}\ \frac{P_2}{Q_2}\ \cdots\ \frac{P_j}{Q_j}\,[\ \frac{p_1}{q_1}\ \frac{p_2}{q_2}\ \cdots\ \frac{p_k}{q_k}\ ]\,.$$

Note that the FRACTRAN-$1\frac{1}{2}$ program

$$m[\ f_1 f_2\ \cdots\ f_k\ ]$$

started at N, simulates the FRACTRAN-1 program

$$[\ f_1 f_2\ \cdots\ f_k\ ]$$

started at mN.

We shall usually suppose tacitly that our FRACTRAN programs are only applied to working numbers N whose prime divisors appear among the factors of the numerators and denominators of the fractions mentioned.

7. Beginners' Guide to FRACTRAN Programming.

It's good practice to write FRACTRAN programs as flowcharts, with a node for each program line and arrows between these nodes marked with the appropriate fractions. We use the different styles of arrowhead

$$\xrightarrow{f} \quad \xrightarrow{f} \quad \xrightarrow{f} \quad \xrightarrow{f}$$

for the options with decreasing priorities from a given node, and if several options with fractions f, g, h at a node have adjacent priorities, we often amalgamate them into a single arrow:

$$\xrightarrow{f,\ g,\ h}$$

The different primes that arise in the numerators and denominators of the various fractions may be regarded as storage registers, and in a state in which the current working integer is

$$N = 2^a\, 3^b\, 5^c\, 7^d \ldots,$$

we say that

register 2 holds a, or $r_2 = a$
register 3 holds b, or $r_3 = b$
register 5 holds c, or $r_5 = c$
register 7 holds d, or $r_7 = d$
etc.

FRACTRAN program lines are then regarded as instructions to change the contents of these registers by various small amounts, subject to the overriding requirement that no register may ever contain a negative number. Thus the line

$$\text{line } 13 : \frac{2}{3} \to 7,\ \frac{4}{5} \to 14$$

either	*replaces*	r_2 *by* $r_2 + 1$, r_3 *by* $r_3 - 1$	(if $r_3 > 0$)
or	*replaces*	r_2 *by* $r_2 + 2$, r_5 *by* $r_5 - 1$	(if $r_5 > 0$)
or	*stops*		(if $r_3 = r_5 = 0$).

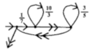
In our figures, unmarked arrows are used when the associated fractions are 1. A tiny incoming arrow to a node indicates that that node will be used as a starting node; a tiny outgoing arrow marks a node that may be used as a stopping node. A few simple examples should convince the reader the FRACTRAN really does have universal computing power. (Readers familiar with Minsky's register machines will see that FRACTRAN can trivially simulate them.)

The program

is a destructive adder: when started with $r_2 = a$, $r_3 = b$, it stops with $r_2 = a + b$, $r_3 = 0$. We can make it less destructive by using register 5 as working space: the program

when started with $r_2 = a$, $r_3 = b$, $r_5 = 0$, stops with $r_2 = a + b$, $r_3 = b$, $r_5 = 0$.

By repeated addition, we can perform multiplication: the program



started with $r_2 = a$, $r_3 = b$, $r_5 = 0$, $r_7 = c$, stops with $r_2 = a + bc$, $r_3 = b$, $r_5 = r_7 = 0$. We add an order $\frac{1}{3}$ ("clear 3") at the starting/finishing node and formulate the result as an official FRACTRAN program:

line 1 : $\frac{1}{7} \to 2,\ \frac{1}{3} \to 1$

line 2 : $\frac{10}{3} \to 2,\ \frac{1}{1} \to 3$

line 3 : $\frac{3}{5} \to 3,\ \frac{1}{1} \to 1.$

When started at line 1 with $N = 3^b 7^c$, it stops at line 1, with $N = 2^{bc}$.

The program obtained by preceding this one by a new

line 0 : $\frac{21}{2} \to 0,\ \frac{1}{1} \to 1$,

when started at line 0 with $N = 2^n$, stops at line 1 with $N = 2^{n^2}$.

8. How to Use the FRACTRAN-1 Model.

You can use a FRACTRAN-1 machine to simulate arbitrary FRACTRAN programs. You must first clear the given program of loops, in a way we explain later, and then label its lines (nodes) with prime numbers $P, Q, R, \ldots$ larger than any of the primes appearing in the numerators and denominators of any of its fractions. The FRACTRAN-1 program simulates

$$\text{line } P : \frac{a}{b} \to Q,\ \frac{c}{d} \to R,\ \frac{e}{f} \to S, \ldots$$

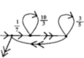
by the fractions

$$\frac{aQ}{bP}\ \ \frac{cR}{dP}\ \ \frac{eS}{fP}\ \ \cdots$$

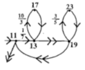
in that order. If the FRACTRAN-0 program when started with N in state P stops with M at line Q , the simulating FRACTRAN-1 program when started a PN stops at QM .

Manufacturer's note. Our guarantee is invalid if you use your FRACTRAN-1 machine in this way to simulate a FRACTRAN program that has loops at several nodes. Such loops may be eliminated by splitting nodes into two.

The third of our examples



becomes



when each of the two nodes with a loop is split in this way, and the new nodes are labeled with the primes 11, 13, 17, 19, 23. Accordingly, it is simulated by the FRACTRAN-1 program

$$[\ \frac{13}{77}\ \frac{170}{39}\ \frac{19}{13}\ \frac{13}{17}\ \frac{69}{95}\ \frac{11}{19}\ ].$$

If started with $N = 2^a\, 3^b\, 7^c\, 11$, this program stops with $N = 2^{a+bc}\, 3^b\, 11$. (The factors of 11 here correspond to the starting and stopping states of the simulated machine.)

We note that it is permissible to label one of the states with the number 1, rather than a large prime number. The fractions corresponding to transitions from this state should be placed (in their proper order) at the *end* of the FRACTRAN-1 program. If this is done, loops, provided they have lower priority than any other transition, are permitted at node 1. Thus the FRACTRAN-1 program

$$[\ \frac{170}{39}\ \frac{19}{13}\ \frac{13}{17}\ \frac{69}{95}\ \frac{1}{19}\ \frac{13}{7}\ \frac{1}{3}\ ]$$

simulates the previous program with a loop order $\frac{1}{3}$ adjoined at the starting/stopping node, which has been relabelled 1. This program, started at $3^b 7^c$, stops at 2^{bc}.

A given FRACTRAN program can always be cleared of loops and adjusted so that 1 is its only stopping node. It follows that we can simulate it by a FRACTRAN-1 program that starts at PN and stops at M when the original program started at N and stopped at M. As we remarked in Section 6, we can simulate this by a FRACTRAN-$1\frac{1}{2}$ program

$$P[\ \cdots\ ]$$

which starts at N and stops at M.

9. Your PIGAME Guarantee.

We now prove Theorem 2, which is equivalent to the assertion that the program

$$[\ \frac{365}{46}\ \frac{29}{161}\ \cdots\ \frac{1}{11}\ \frac{1}{1024}\ ]$$

(obtained by ignoring factors of 97 and dropping the final fraction $\frac{89}{1}$ of PIGAME), when started at $2^n \cdot 89$, stops at $2^{\pi(n)}$. This FRACTRAN-1 program has been obtained from the FRACTRAN program of Figure 3 by the method outlined in the last section. The pairs of nodes 13 & 59, 29 & 71, 23 & 73, 31 & 67, and 43 & 53 were originally single nodes with loops.

We shall only sketch the action of this program, which we separate into three phases. The first phase ends when the program first reaches node 37, the second phase when it first reaches node 41, and the third phase when it finally stops, at node 1.

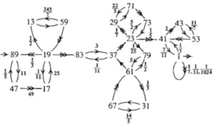


Figure 3. A FRACTRAN program for digits of π .

The first phase, started at 89 with register contents

$$r_2 = n, \ r_3 = r_5 = r_7 = r_{11} = 0,$$

reaches 37 with contents

$$r_2 = 0, \ r_3 = 1, \ r_5 = E, \ r_7 = 2 \cdot 10^n, \ r_{11} = 0,$$

where E is a very large even number. To see this, ignore the 5 and 11 registers for a moment, and see that it initially sets $r_7 = 2$. Then each pass around the triangular region multiplies r_7 by 5 and puts it into r_3 and is followed by passes around the square region which double r_3 and put it back into r_7. This is done n times, so that at the end of this phase we have $r_7 = 2 \cdot 10^n$, as desired.

The first pass around the square ends with 4 in r_5, and each subsequent pass at least doubles this number, while keeping it even. At the last stage we pass around this region 10^n times and finish with an even number $E \geq 4 \times 2^{10^n}$ in r_5 . It's easy to check that registers 2, 3, and 11 end with the indicated values.

At the end of the second phase, we shall have

$$r_2 = r_5 = r_7 = 0\,,$$

$$r_3 = 2 \times 10^n \times E(E-2)(E-2)(E-4)(E-4)(E-6) \cdots 4 \cdot 4 \cdot 2 \cdot 2 \stackrel{\Delta}{=} N\,,$$

$$r_{11} = 1 \times (E-1)(E-1)(E-3)(E-3)(E-5)(E-5) \cdots 5 \cdot 3 \cdot 3 \cdot 1 \stackrel{\Delta}{=} D\,.$$

This is fairly easy to check, the essential point being that each sojourn in the upper region multiplies r_7 by r_5 and puts it into r_{11} (preserving the value of r_5 but clearing r_7), while in the lower region, we multiply r_3 by r_5 into r_7 in a similar way, and then (at the left) transfer r_{11} back to r_3. Register 5 is decreased by 1 as we pass from the upper to the lower region; but when $r_5 = 1$ we instead clear it and pass to node 41, entering the third phase.

Now *Wallis' product* is

$$\frac{\pi}{2} = \frac{2}{1}\,\frac{2}{3}\,\frac{4}{3}\,\frac{4}{5}\,\frac{6}{5}\,\frac{6}{7}\,\frac{8}{7}\,\frac{8}{9}\,\frac{10}{9}\,\frac{10}{11} \cdots,$$

in which the successive fractions are obtained by alternately increasing the denominator and numerator. If we truncate it so as only to include all factors whose numerator and denominator are at most K, we obtain an approximation π_K for π which is within at most $\frac{\pi}{K}$ of π. So our $\frac{N}{D} = 10^n \cdot \pi_E$, where π_E is a very good approximation indeed to π. It is in fact so good that the nth decimal digit of π_E is the same as that of π. This digit can be obtained by reducing the integer part of $\frac{N}{D}$ modulo 10, and it is easy to check that the third phase of our program does just this, putting the answer in register 2 and clearing all other registers.

The assertion about the nth decimal digit of π_E is not trivial. For $n = 0$, our approximation π_E is $\pi_4 = \frac{32}{9}$. For $n = 1$ or 2, we have $|\pi_E - \pi| < \frac{\pi}{4 \times 2^{10}}$ which is less than $\frac{1}{1000}$, and since $\pi = 3.141\cdots$

the nth digits ($n = 1$ and 2) after the decimal point in π_E must both be correct.

For $n \geq 3$, the error in π_E is at most

$$\frac{\pi}{4 \times 2^{10^n}} < \frac{1}{(1000)^{10^{n-1}}} = 10^{-3 \times 10^{n-1}} < 10^{-42n}.$$

The desired assertion now follows from Mahler's [4] famous irrationality measure for π: if $\frac{p}{q}$ (in least terms) is any nonintegral rational number, then

$$\left| \pi - \frac{p}{q} \right| > \frac{1}{q^{42}}.$$

10. How to Use Our Universal Program.

In this section, we prove Theorem 3, using an ingenious lemma due to John Rickard. We shall call a FRACTRAN-1 program $[f_1, f_2, \ldots, f_k]$ *monotone* if $f_1 < f_2 < f_3 < \cdots < f_k$.

Lemma: Any FRACTRAN-1 program can be simulated by a monotone one that starts and stops with the same numbers.

Proof. Choose a new prime P that is bigger than the ratio between any two of the f_i and bigger than the inverse of any f_i. Then $[\frac{1}{P}, Pf_1, P^2f_2, P^3f_3, \ldots, P^kf_k]$ simulates $[f_1, f_2, f_3, \ldots, f_k]$ and is monotone. The new program behaves exactly like the old one, except that at each step a power of P is introduced, only to be immediately cleared away before we copy the next step.

We shall call a FRACTRAN-$1\frac{1}{2}$ program

$$\dot{f}_1, \dot{f}_2, \ldots, \dot{f}_j \, [f_1, f_2, \ldots, f_k]$$

monotone if

$$\dot{f}_1 < \dot{f}_2 < \cdots < \dot{f}_j \text{ and } f_1 < f_2 < \cdots < f_k.$$

Then our universal program simulates monotone FRACTRAN-1$\frac{1}{2}$ programs. It codes such a program by three numbers, M^*, M, and d, defined as follows.

We take d to be any common denominator of all the fractions mentioned and suppose the given FRACTRAN-1$\frac{1}{2}$ program is

$$\frac{m_1^*}{d}\ \frac{m_2^*}{d}\ \cdots\ \frac{m_j^*}{d}\ [\ \frac{m_1}{d}\ \frac{m_2}{d}\ \cdots\ \frac{m_k}{d}\ ].$$

We then adjoin dummy numbers m_{j+1}^* and m_{k+1}, which are both multiples of d and which satisfy

$$m_1^* < m_2^* < \cdots < m_j^* < m_{j+1}^*, \quad m_1 < m_2 < \cdots < m_k < m_{k+1},$$

and $[\ \frac{1}{2} M^*\ ] \le M$

where

$$M^* = 2^{m_1^*} + 2^{m_2^*} + \cdots + 2^{m_{j+1}^*}$$

$$M = 2^{m_1} + 2^{m_2} + \cdots + 2^{m_{k+1}}.$$

The universal program POLYGAME, started at

$$2^N\, 3^M\, 5^{M^*}\, 17^{d-1}\, 23$$

will simulate the given FRACTRAN-1$\frac{1}{2}$ program, started at N. This universal FRACTRAN-1 program was obtained from the FRACTRAN program shown in Figure 4, and accordingly, we consider starting the latter with $r_2 = N$, $r_3 = M$, $r_5 = M^*$, $r_{17} = d-1$, at the node 23.

This works roughly as follows. After a new N has been found, the program computes successive multiples $N, 2N, 3N, \ldots, mN$, and simultaneously repeatedly halves M to get $[\ M/2\ ]$, $[M/4\ ]$, ..., $[\ M/2^m\ ]$. If $[\ M/2^m\ ]$ is odd, so that m is one of the m_i, it sees whether Nm is a multiple of d, and if so resets M and takes a new $N = mN/d$, *unless* m was m_{k+1} (i.e., $[\ M/2^m\ ] = 1$), when it arranges to stop at node 1 with register 2 containing N and all other registers empty. For the first pass, it uses M^* in place of M.

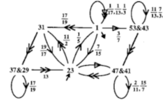


Figure 4. A flowchart for POLYGAME.

Registers 13, 17, 19 function as a counter, whose count is stored in a form from which we can see at once if it is a multiple of d. If

$$r_{13} = q,\ r_{19} = r,\ r_{17} = d - 1 - r, \text{ with } 0 \le r < d,$$

then *the count* is the number $qd + r$. If the machine arrives at node 31 ("enters the counter") with these values, then when it next arrives at node 23 ("leaves the counter"), we shall have

$$r_{13} = q,\ r_{19} = r + 1,\ r_{17} = d - 1 - (r + 1), \text{ if } r < d - 1,$$

$$r_{13} = q + 1,\ r_{19} = 0,\ r_{17} = d - 1, \text{ if } r = d - 1.$$

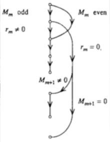
In other words, the value of the count will have increased by 1.

So if the machine is started at 23, with $r_5 = r_{11} = 0$ and $r_2 = N$, it will increase the count by N while transferring N from register 2 to register 11, and then go to node 47 (where its first action will be to retransfer N from register 11 back to register 2).

Table 2. The Action of POLYGAME

node	2	3	5	7	11	13	17	19
	Contents of registers:							
23	N	M	M_m	0	0	q_m	$d-1-r_m$	r_m
1	N	$M-M_{m+1}$	0	M_{m+1}	0	q_m	$d-1-r_m$	r_m
23	N	$M-M_{m+1}$	0	M_{m+1}	0	q_m	$d-1-r_m$	r_m
47 & 41	0	$M-M_{m+1}$	0	M_{m+1}	N	q_{m+1}	$d-1-r_{m+1}$	r_{m+1}
23	N	M	M_{m+1}	0	0	q_{m+1}	$d-1-r_{m+1}$	r_{m+1}
47 & 41	0	0	0	M	$\frac{mN}{d}$	0	$d-1$	0
23	$\frac{mN}{d}$	M	M	0	0	0	$d-1$	0
1	N	0	0	0	0	0	0	0

action



$$mN = q_m \cdot d + r_m \ (0 \le r_m < d) \qquad M_m = [\, M_0 / 2^m \,]$$

After these remarks, the reader should have little difficulty in verifying the transitions between particular configurations shown in Table 2.

We suppose that for particular positive numbers d, N, M, and M_0 with $[\frac{1}{2} M_0] \leq M$ we define for varying values of m the numbers M_m, q_m, r_m by

$$M_m = [M_0/2^m]$$

$$mN = q_m d + r_m \quad (0 \leq r_m < d).$$

Then Table 2 shows that unless M_m is odd and $r_m = 0$, the special type of configuration in the first line of the table leads to a similar one (in the fifth line) with m increased by 1. In the excepted case, if $M_{m+1} \neq 0$, we obtain another such special configuration (in the seventh line), but with m (and the count) reset to 0, the new initial value $M_0 = M$ for M_m, and $\frac{mN}{d}$ as the new N. If instead M_{m+1} was 0, we arrive at the last line of the table, and *stop* at node 1, with N in register 2 and all other registers empty. The cases with M_m odd and $r_m = 0$ are called *resets*.

Now suppose we start the machine in the special configuration in the top line of the table, with $m = 0$, and the initial value M_0 of M_m set to the number

$$2^{m_0} + 2^{m_1} + \cdots + 2^{m_{k+1}},$$

where

$$m_0 < m_1 < \cdots < m_{k+1}$$

and m_{k+1} is divisible by d. Then before the next reset, we have the equivalences

$$\begin{aligned} M_m \text{ odd} &\iff m \text{ is one of the } m_i \\ r_m = 0 &\iff mN/d \text{ is an integer} \\ M_{m+1} = 0 &\iff m = m_k. \end{aligned}$$

So the next reset will be at the first of the m_i for which $m_i N/d$ is integral, and will *either* replace N by $m_i\, N/d$, and reset m to 0 and M_m to M (if $i < k$), or *stop* at node 1, with N in register 2 and the rest empty $(i = k)$.

This completes the required verifications. Initially, we set $m = 0$ and $M_0 = M^*$, but all subsequent resets will put $M_0 = M$, in accordance with the rules for FRACTRAN-$1\frac{1}{2}$ programs.

A FRACTRAN-1 program is a FRACTRAN-$1\frac{1}{2}$ program with $M = M^*$. For this we can use the alternate catalogue number $7^M\, 17^{d-1}\, 41$.

11. Applications, Improvements, Acknowledgments.

For the function

$$g(N) = \begin{cases} \frac{1}{2} N & (N \text{ even}) \\ 3N + 1 & (N \text{ odd}), \end{cases}$$

the *Collatz problem* asks whether for every positive integer N there exists a k for which $g^k(N) = 1$. See [3] for a survey of this problem.

We can ask similar questions for more general *Collatz functions*

$$g(N) = a_N\, N + b_N\,,$$

where a_N and b_N are rational numbers that only depend on the value of N modulo some fixed number D. We proved in [1] that there is no algorithm for solving arbitrary Collatz problems. Indeed, for any computable function $f(n)$, there is a FRACTRAN-1 program $[f_1\, f_2 \cdots f_k]$ with the property that when we start it at 2^n, the first strictly later power of 2 will be $2^{f(n)}$. In other words, we can define f by

$$2^{f(n)} = g^k\,(2^n),$$

where k is the smallest positive integer for which $g^k\,(2^n)$ is a power of 2, and the function $g(N)$, which has the above form, is just $f_i\, N$ for the least i which makes this an integer. This result is an explicit version of *Kleene's Normal Form Theorem*.

We note that $g(N)/N$ is a periodic function with rational values, so that $g(N)$ is a Collatz function for which b_N is always 0. So even for Collatz functions of this special type there can be no decision procedure. By applying the argument to a universal fraction game, we can get a *particular* Collatz-type problem with no decision procedure.

(We remark that of course Collatz problems with arbitrary b_N are harder to solve, rather than easier. We might, for instance, define one that simulates a program written in 10 segments, each segment using only the numbers ending in a given decimal digit, and in which control is transferred between the segments only at certain crucial--and recursively unpredictable--times.)

John Rickard tells me that he has found a seven fraction universal program of type $2^{2^n} \cdot c \to 2^{2^{f(n)}}$ and a nine fraction one of type $2^n \cdot c \to 2^{f(n)}$. However, it seems that his fractions are much too complicated ever to be written down. I used one of Rickard's ideas in Section 10. Mike Guy gave valuable help in computing the catalogue numbers in Section 2. Of course, the responsibility for any errors in these numbers rests entirely with him.

REFERENCES

[1] J.H. Conway, "Unpredictable Iterations," in Proceedings of the Number Theory Conference, Boulder, Colorado, pp. 49-52 (1972).

[2] J.H. Conway, "FRACTRAN - A Simple Universal Programming Language for Arithmetic," *Open Problems Commun. Comput.*, pp. 4-26 (1986).

[3] J.C. Lagarias, "The $3x + 1$ Problem and Its Generalizations," *Am. Math. Monthly*, 92, No. 1, pp. 3-25 (1985).

[4] K. Mahler, "On the Approximation of π," *Indagnationes Math.*, 15, pp. 30-42 (1953).