Stern-Brocot tree

There's a beautiful way to construct the set of all nonegative fractions m/n with $m \perp n$(It means gcd(m,n)=1), called the Stern-Brocot tree because it was discovered independently by Moritz Stern, a German mathematician, and Achillle Brocot, a French clockmacker. The idea is to start with the two fractions $\left(\frac{0}{1}, \frac{1}{0}\right)$ and then to repeat the following operation as many times as desired:

$$Insert \ \frac{m+m'}{n+n'} \ between \ two \ adjacent \ fractions \ \frac{m}{n} \ and \ \frac{m'}{n'} .$$

The new fraction $\frac{m+m'}{n+n'}$ is called the mediant of m/n and m'/n'. For example, the first step gives us one new entry between $\frac{0}{1}$ and $\frac{1}{0}$,

$$\frac{0}{1}, \frac{1}{1}, \frac{1}{0};$$

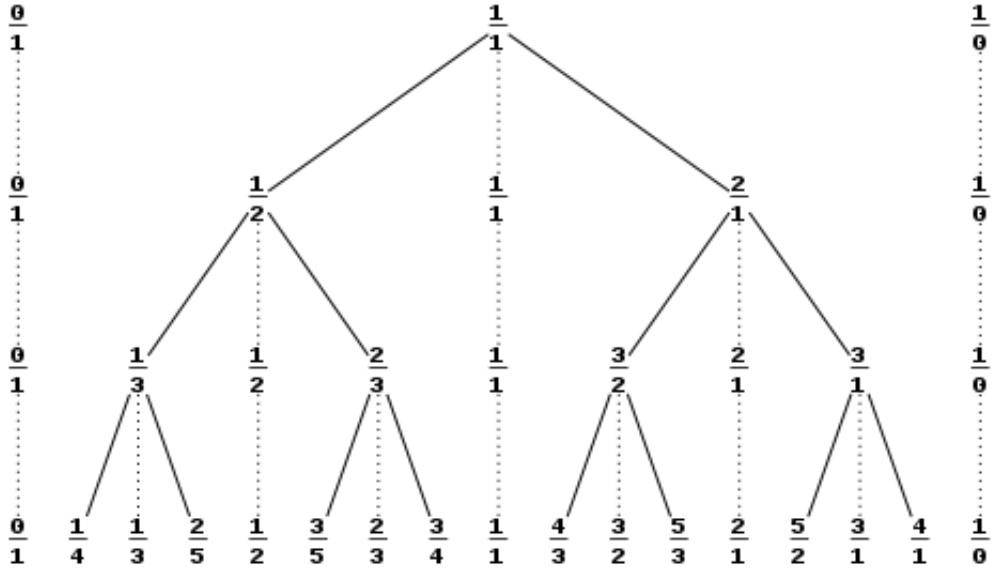
and The next gives two more:

$$\frac{0}{1}, \frac{1}{2}, \frac{1}{1}, \frac{2}{1}, \frac{1}{0};$$

The next gices four more,

$$\frac{0}{1}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{1}{1}, \frac{3}{2}, \frac{2}{1}, \frac{3}{1}, \frac{1}{0};$$

and then we'll get 8, 16, and so on. The entire array can be regarded as an infinite binary tree structure whose top levels look like this:



Each fraction is $\frac{m+m'}{n+n'}$, Where $\frac{m}{n}$ is the nearest ancestor above and to the left, and $\frac{m'}{n'}$ is the nearest ancestor aboce and to rhe right. Many patterns can be observed in this tree.

Why does this construction work?Why, for example does each mediant fraction $(m+m')/(n+n')$ turn out to be in lowest terms when it appears in this tree?(IF m,m',n,n'were all odd, we'd get even/even; somehow the construction guarantees that fractions with odd numerators and denomiators never appear next to each other.) And why do all possible fraction m/n occur exactly once? Why can't a particular fraction occur twice,or not at all?

All of these questions have amazingly simple answers1, based on the following fundamental fact: If m/n and m'/n' are consecutive factions at any stage of the construction, we have

$$m'n - mn' = 1.$$

This relations is true initially($1 * 1 - 0 * 0 = 1$); and when we insert a new mediant $(m+m')/(n+n')$, the new cases that need to be checked are

$$(m+m')n - m(n+n') = 1$$
$$m'(n+n') - (m+m') = 1.$$

Both of these equation are equivalent to the original condition $m'n - mn' = 1$ that they replace. Therefore it is invariant at all stages of the construction.

Furthermore, if $m/n < m'n'$ and if all values are nonegative, it's easy to verify that

$$m/n < (m+m')/(n+n') < m'/n'.$$

A mediant fraction isn't halfway between its progenitors, but it does lie somewhere in between. Therefore the construction preserves order, and we couldn't possibly get the same fraction in two different places.

One question stiil remains. Can any positive fraction a/b with $a \perp b$ possibly be omitted? The answer is no, because we can confine the construction to the immediate neighberhood of a/b, and in this region the behavior is easy to analyseA: Initially we have

$$\frac{m}{n} = \frac{0}{1} < (\frac{a}{b}) < \frac{1}{0} = \frac{m'}{n'}$$

where we put parentheses around $\frac{a}{b}$ to indicate that it's not really present yet. Then if at some stage we have

$$\frac{m}{n} < (\frac{a}{b}) < \frac{m'}{n'},$$

the construction forms $(m+m')/(n+n')$ and there are three cases. Either $(m+m')/(n+n') = a/b$ and we win; or $(m+m')/(n+n')a/b$ and we can set $m' \leftarrow m+m', n' \leftarrow n+n'$. This process cannot go on indefinitely, because the conditions

$$\frac{a}{b} - \frac{m}{n} > 0 \qquad and \qquad \frac{m'}{n'} - \frac{a}{b} > 0$$

imply that

$$an - bm \geq 1 \qquad and \qquad bm' - an' \geq 1;$$

hence

$$(m' + n')(an - bn) + (n+m)(bm' - an') \geq m' + n' + m + n;$$

and this is the same as $a + b \geq m' + n' + m + n$ by $m'n - mn' = 1$. Either m or n or m' or n' invreases at each step, so we must win after at most $a + b$ steps.

The *Farey series* of order N, denoted by F_N, is the set of all reduced fractions between 0 and 1 whose denominators are N or less, arranged in increasing order. For example, if $N = 6$ we have

$$F_6 = \frac{0}{1}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5} \frac{5}{6}, \frac{1}{1}.$$

We can obtain obtain F_N in general by starting with $F_1 = \frac{0}{1}, \frac{1}{1}$ and then insertin mediants whenever it's possible to do so without getting a denominator is > N.

This method of construction reveals that F_N can be obtained in a simple way from F_{N-1}; We simply insert the fraction $(m + m')/N$ between consecutive fractions $m/n, m'/n'$ of F_{N-1} whose denominators sum to N. For example, it's easy to obtain F_7 from the elements of F_6, by inserting $\frac{1}{7}, \frac{2}{7}, \cdots, \frac{6}{7}$ to the state rule:

$$F_7 = \frac{0}{1}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{2}{7}, \frac{1}{3}, \frac{2}{5}, \frac{3}{7}, \frac{1}{2}, \frac{4}{7}, \frac{3}{5}, \frac{2}{3}, \frac{5}{7}, \frac{3}{4}, \frac{4}{5}, \frac{5}{6}, \frac{6}{7}, \frac{1}{1}.$$

When N is prime, N-1 new fractions will appear; but otherwise we'll have fewer than N - 1, because this process generates only numerators that are relatively primes to N.
We can, in fact, regard the Stern-Brocot tree as a *number system* for representing rational numbers, because each positive, reduced fraction occurs exactly once. Let's use the letters L and R to stand for going down to the fraction; then a string of L's and R's uniquely identifies a place in the tree. For example, LRRL means that we fo left from $\frac{1}{1}$ down to $\frac{1}{2}$, then right to $\frac{2}{3}$, then right to $\frac{3}{4}$, then left to $\frac{5}{7}$. We can consider LRRL to be a representation of $\frac{5}{7}$. Every positive fraction gets represented in this way as a unique string of L's and R's.

Well, actually there's a slight problem: The fraction $\frac{1}{1}$ corresponds to the empty string, and we need a notation for that. Let's agree to call it I, because that looks something like 1 and it stands for "identity".

This representation raises two natural questions: (1) Given positive integers m and n with $m \perp n$, what is the string of L's and R's that corresponds to m/n? (2) Given a string of L's ans R's, what fraction corresponds to it? Question 2 seems easier, so let's work on it first. We define

$$f(S) = fraction\ corresponding\ to\ S$$

When S is a string of L's ans R's. For example, $f(LRRL) = \frac{5}{7}$.

According to the construction, $f(S) = (m + m')/(n + n')$ if m/n and m'/n' are the closest fractions preceding and following S in the upper levels of the tree. Initially $m/n = 0/1$ and $m'/n' = 1/0$; then we successively replace either m/n or m'/n' by the mediant $(m' + m)/n' + n$ as we move right or left in the tree, respectively.

How can we capture this behavior in mathematical formulas that are easy to deal with? A bit of experimentation suggests that the best way is to maintain a 2X2 matrix

$$M(S) = \begin{pmatrix} n & n' \\ m & m' \end{pmatrix}$$

that holds the four quantities involved in the ancestral fractions m/n and m'/n' enclosing f(S). We could put the m's on top and the n's on the bottom, fractionwise; but upside-down arrangedment works out more nicely because we have $M(1) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ when the process starts, and $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ is traditionally called the identity matrix I.

A step to the left replace n' by n+n' and m' by m+m'; hence

$$M(SL) = \begin{pmatrix} n & n+n' \\ m & m+m' \end{pmatrix} = \begin{pmatrix} n & n' \\ m & m' \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = M(S)\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Simularly it turns out that

$$M(SR) = \begin{pmatrix} n+n' & n' \\ m+m' & m' \end{pmatrix} = M(S)\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

Therefore if we define L and R as 2×2 matrices,

$$L = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \qquad R = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}.$$

we get the simple formula $M(S) = S$, by induction on the length of S. Isn't that nice? For example,

$$M(LRRL) = LRRL = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 4 \\ 2 & 3 \end{pmatrix};$$

the ancestral fractions that enclose LRRL = $\frac{5}{7}$ are $\frac{2}{3}$ and $\frac{3}{4}$. And this construction gives us the answer to Question 2:

$$f(S) = f\left(\begin{pmatrix} n & n' \\ m & m' \end{pmatrix}\right) = \frac{m+m'}{n+n'}.$$

How about Question 1? That's easy, now that we understand the fundamental connection between tree nodes and 2×2 matrices. Given a pair of positive integers m and n with $m \perp n$, we can find the position of m/n in the Stern-Brocot tree by "binary search" as follows:

```
1.  S := 1
2.  while m / n != f(S) do
3.      if m/n < f(S) then (output(L); S := SL)
4.                    else (output(R); S := SR).
```

This outputs the desired strng of L's and R's.

There's also another way to do the same job, by changing m and n instead of maintaining the state S. If S is any 2×2 matrix, we have

$$f(RS) = f(S) + 1$$

because RS is like S but with the top row added to the bottom row.

$$S = \begin{pmatrix} n & n' \\ m & m' \end{pmatrix}; \qquad RS = \begin{pmatrix} n & n' \\ m+n & n'+m' \end{pmatrix};$$

If we carry out the binary search algorithm on a fraction m/n with $m > n$, the first output will be R; hence the subsequent behavior of the algorithm will have f(S) exactly 1 greater than if we had begun with $(m-n)/n$ instead of m/n. A similar property holds for L, and we have

$$\frac{m}{n} = f(RS) \qquad \Leftrightarrow \qquad \frac{m-n}{n} = f(S), whenm > n;$$
$$\frac{m}{n} = f(RS) \qquad \Leftrightarrow \qquad \frac{m}{n-m} = f(S), whenm < n.$$

This means that we can transform the binary search algorithm to the following matrix-free procedure:

```
1.  while m != n do
2.      if m < n then (output(L); n := n - m)
3.               else (output(R); m := m - n)
```

For example, given $m/n = 5/7$, we have successively

$m = 5$	5	3	1	1
$n = 7$	2	2	2	1
output	L	R	R	L

in the simplified algorithm.

$$e = RL^0RLR^2LRL^4RLR^6LRL^8RLR^{10}LRL^{12}RL\cdots;$$