Last update: April 15, 2025 Translated From: e-maxx.ru

The Stern-Brocot tree and Farey sequences

Stern-Brocot tree

The Stern-Brocot tree is an elegant construction to represent the set of all positive fractions. It was independently discovered by German mathematician Moritz Stern in 1858 and by French watchmaker Achille Brocot in 1861. However, some sources attribute the discovery to ancient Greek mathematician Eratosthenes.

The construction starts at the zeroth iteration with the two fractions

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

where it should be noted that the second quantity is not strictly a fraction, but it can be interpreted as an irreducible fraction representing infinity.

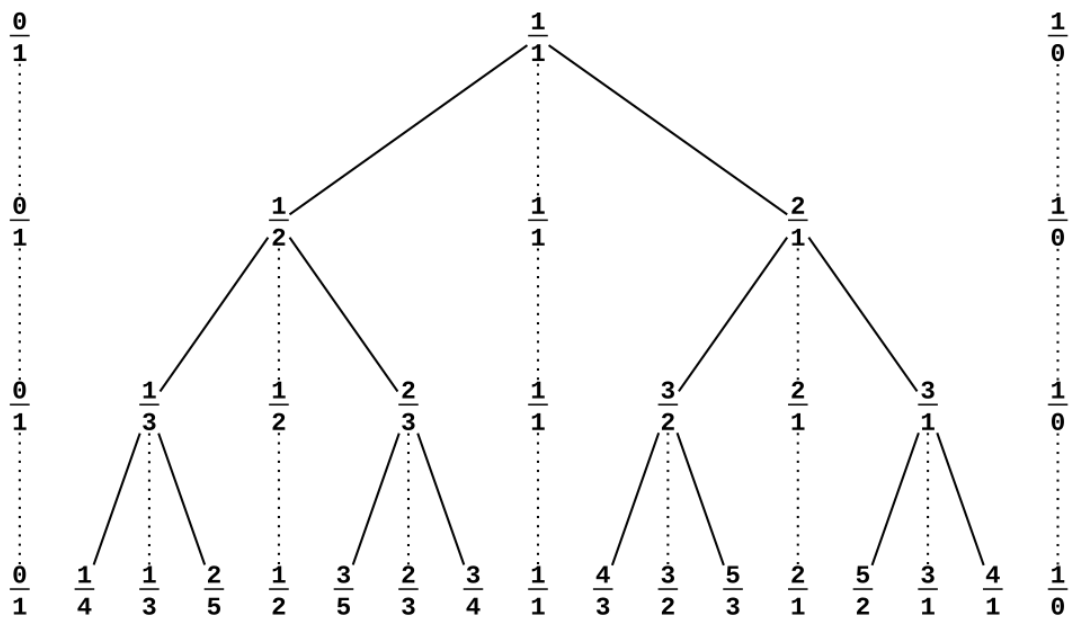
At every subsequent iteration, consider all adjacent fractions $\frac{a}{b}$ and $\frac{c}{d}$ and insert their mediant $\frac{a+c}{b+d}$ between them.

The first few iterations look like this:

$$\begin{gathered}\frac{0}{1}, \frac{1}{1}, \frac{1}{0}\\ \frac{0}{1}, \frac{1}{2}, \frac{1}{1}, \frac{2}{1}, \frac{1}{0}\\ \frac{0}{1}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{1}{1}, \frac{3}{2}, \frac{2}{1}, \frac{3}{1}, \frac{1}{0}\end{gathered}$$

Continuing this process to infinity this covers *all* positive fractions. Additionally, all fractions will be *unique* and *irreducible*. Finally, the fractions will also appear in ascending order.

Before proving these properties, let us actually show a visualization of the Stern-Brocot tree, rather than the list representation. Every fraction in the tree has two children. Each child is the mediant of the closest ancestor on the left and closest ancestor to the right.



Proofs

Ordering. Proving ordering is simple. We note that the mediant of two fractions is always in-between the fractions

$$\frac{a}{b} \leq \frac{a+c}{b+d} \leq \frac{c}{d}$$

given that

$$\frac{a}{b} \leq \frac{c}{d}.$$

The two inequalities can be easily shown by rewriting the fractions with common denominators.

As the ordering is ascending in the zeroth iteration, it will be maintained at every subsequent iteration.

Irreducibility. To prove this we will show that for any two adjacent fractions $\frac{a}{b}$ and $\frac{c}{d}$ we have that

$$bc - ad = 1.$$

Recall that a Diophantine equation with two variables $ax + by = c$ has a solution iff c is a multiple of $\gcd(a, b)$. In our case this implies that $\gcd(a, b) = \gcd(c, d) = 1$, which is what we want to show.

Clearly at the zeroth iteration $bc - ad = 1$. What remains to be shown is that mediants retain this property.

Assume our two adjacent fractions uphold $bc - ad = 1$, after the mediant is added to the list

$$\frac{a}{b}, \frac{a+c}{b+d}, \frac{c}{d}$$

the new expressions become

$$\begin{aligned} b(a+c) - a(b+d) &= 1 \\ c(b+d) - d(a+c) &= 1 \end{aligned}$$

which, using that $bc - ad = 1$, can be easily shown to be true.

From this we see that the property is always maintained and thus all fractions are irreducible.

The presence of all fractions. This proof is closely related to locating a fraction in the Stern-Brocot tree. From the ordering property we have that left subtree of a fraction contains only fractions smaller than the parent fraction, and the right subtree contains only fractions larger than the parent fraction. This means we can search for a fraction by traversing the tree from the root, going left if the target is smaller than the fraction and going right if the target is larger.

Pick an arbitrary positive target fraction $\frac{x}{y}$. It is obviously between $\frac{0}{1}$ and $\frac{1}{0}$, so the only way for the fraction to not be in the tree is if it takes an infinite number of steps to get to it.

If that is the case we would at all iterations have

$$\frac{a}{b} < \frac{x}{y} < \frac{c}{d}$$

which (using the fact than an integer $z > 0 \iff z \geq 1$) can be rewritten as

$$\begin{aligned} bx - ay &\geq 1 \\ cy - dx &\geq 1. \end{aligned}$$

Now multiply the first inequality by $c + d$ and the second with $a + b$ and add them to get

$$(c+d)(bx - ay) + (a+b)(cy - dx) \geq a + b + c + d.$$

Expanding this and using the previously shown property $bc - ad = 1$ we get that

$$x + y \geq a + b + c + d.$$

And given that at every iteration at least one of a, b, c, d will increase, the fraction searching process will contain no more than $x + y$ iterations. This contradicts the assumption that the path to $\frac{x}{y}$ was infinite and hence $\frac{x}{y}$ must be part of the tree.

Tree Building Algorithm

To build any subtree of the Stern-Brocot tree, it suffices to know the left and right ancestor. On the first level, the left and right ancestors are $\frac{0}{1}$ and $\frac{1}{0}$ respectively. Using these, we calculate the mediant and proceed one level deeper, with the mediant replacing the right ancestor in the left subtree, and vice versa.

This pseudocode tries to build the entire infinite tree:

```
void build(int a = 0, int b = 1, int c = 1, int d = 0, int level = 1) {
    int x = a + c, y = b + d;

    ... output the current fraction x/y at the current level in the tree

    build(a, b, x, y, level + 1);
    build(x, y, c, d, level + 1);
}
```

Fraction Search Algorithm

The search algorithm was already described in the proof that all fractions appear in the tree, but we will repeat it here. The algorithm is a binary search algorithm. Initially we stand at the root of the tree and we compare our target with the current fraction. If they are the same we are done and stop the process. If our target is smaller we move to the left child, otherwise we move to the right child.

Naive search

Here is an implementation that returns the path to a given fraction $\frac{p}{q}$ as a sequence of `'L'` and `'R'` characters, meaning traversal to the left and right child respectively. This sequence of characters uniquely defines all positive fractions and is called the Stern-Brocot number system.

```
string find(int p, int q) {
    int pL = 0, qL = 1;
    int pR = 1, qR = 0;
    int pM = 1, qM = 1;
    string res;
    while(pM != p || qM != q) {
        if(p * qM < pM * q) {
            res += 'L';
            tie(pR, qR) = {pM, qM};
        } else {
            res += 'R';
            tie(pL, qL) = {pM, qM};
        }
        tie(pM, qM) = pair{pL + pR, qL + qR};
    }
    return res;
}
```

Irrational numbers in the Stern-Brocot number system corresponds to infinite sequences of characters. Along the endless path towards the irrational number the algorithm will find reduced fractions with gradually increasing denominators that provides increasingly better approximations of the irrational number. So by taking a prefix of the infinite sequence approximations with any desired precision can be achieved. This application is important in watch-making, which explains why the tree was discovered in that domain.

Note that for a fraction $\frac{p}{q}$, the length of the resulting sequence could be as large as $O(p+q)$, for example when the fraction is of form $\frac{p}{1}$. This means that the algorithm above **should not be used, unless this is an acceptable complexity**!

Logarithmic search

Fortunately, it is possible to enhance the algorithm above to guarantee $O(\log(p+q))$ complexity. For this we should note that if the current boundary fractions are $\frac{p_L}{q_L}$ and $\frac{p_R}{q_R}$, then by doing a steps to the right we move to the fraction $\frac{p_L+ap_R}{q_L+aq_R}$, and by doing a steps to the left, we move to the fraction $\frac{ap_L+p_R}{aq_L+q_R}$.

Therefore, instead of doing steps of `L` or `R` one by one, we can do k steps in the same direction at once, after which we would switch to going into other direction, and so on. In this way, we can find the path to the fraction $\frac{p}{q}$ as its run-length encoding.

As the directions alternate this way, we will always know which one to take. So, for convenience we may represent a path to a fraction $\frac{p}{q}$ as a sequence of fractions

$$\frac{p_0}{q_0}, \frac{p_1}{q_1}, \frac{p_2}{q_2}, \dots, \frac{p_n}{q_n}, \frac{p_{n+1}}{q_{n+1}} = \frac{p}{q}$$

such that $\frac{p_{k-1}}{q_{k-1}}$ and $\frac{p_k}{q_k}$ are the boundaries of the search interval on the k-th step, starting with $\frac{p_0}{q_0} = \frac{0}{1}$ and $\frac{p_1}{q_1} = \frac{1}{0}$. Then, after the k-th step we move to a fraction

$$\frac{p_{k+1}}{q_{k+1}} = \frac{p_{k-1} + a_k p_k}{q_{k-1} + a_k q_k},$$

where a_k is a positive integer number. If you're familiar with continued fractions, you would recognize that the sequence $\frac{p_i}{q_i}$ is the sequence of the convergent fractions of $\frac{p}{q}$ and the sequence $[a_1; a_2, \dots, a_n, 1]$ represents the continued fraction of $\frac{p}{q}$.

This allows to find the run-length encoding of the path to $\frac{p}{q}$ in the manner which follows the algorithm for computing continued fraction representation of the fraction $\frac{p}{q}$:

```
auto find(int p, int q) {
    bool right = true;
    vector<pair<int, char>> res;
    while(q) {
        res.emplace_back(p / q, right ? 'R' : 'L');
        tie(p, q) = pair{q, p % q};
        right ^= 1;
    }
    res.back().first--;
    return res;
}
```

However, this approach only works if we already know $\frac{p}{q}$ and want to find its place in the Stern-Brocot tree.

On practice, it is often the case that $\frac{p}{q}$ is not known in advance, but we are able to check for specific $\frac{x}{y}$ whether $\frac{x}{y} < \frac{p}{q}$.

Knowing this, we can emulate the search on Stern-Brocot tree by maintaining the current boundaries $\frac{p_{k-1}}{q_{k-1}}$ and $\frac{p_k}{q_k}$, and finding each a_k via binary search. The algorithm then is a bit more technical and potentially have a complexity of $O(\log^2(x+y))$, unless the problem formulation allows you to find a_k faster (for example, using `floor` of some known expression).

Farey Sequence

The Farey sequence of order n is the sorted sequence of fractions between 0 and 1 whose denominators do not exceed n.

The sequences are named after English geologist John Farey, who in 1816 conjectured that any fraction in a Farey sequence is the mediant of its neighbors. This was proven some time later by Cauchy, but independent of both of them, the mathematician Haros had come to almost the same conclusion in 1802.

The Farey sequences have many interesting properties on their own, but the connection to the Stern-Brocot tree is the most obvious. In fact, the Farey sequences can be obtained by trimming branches from the tree.

From the algorithm for building the Stern-Brocot tree, we get an algorithm for the Farey sequences. Start with the list of fractions $\frac{0}{1}, \frac{1}{0}$. At every subsequent iteration, insert the mediant only if the denominator does not exceed n. At some point the list will stop changing and the desired Farey sequence will have been found.

Length of a Farey Sequence

A Farey sequence of order n contains all elements of the Farey sequence of order $n-1$ as well as all irreducible fractions with denominator n, but the latter is just the totient $\varphi(n)$. So the length L_n of the Farey sequence of order n is

$$L_n = L_{n-1} + \varphi(n)$$

or equivalently, by unraveling the recursion we get

$$L_n = 1 + \sum_{k=1}^{n} \varphi(k).$$

Contributors:

algmyr (50.86%) adamant-pwn (29.74%) AbhijeetKrishnan (16.38%) jakobkogler (1.72%) mhayter (1.29%)