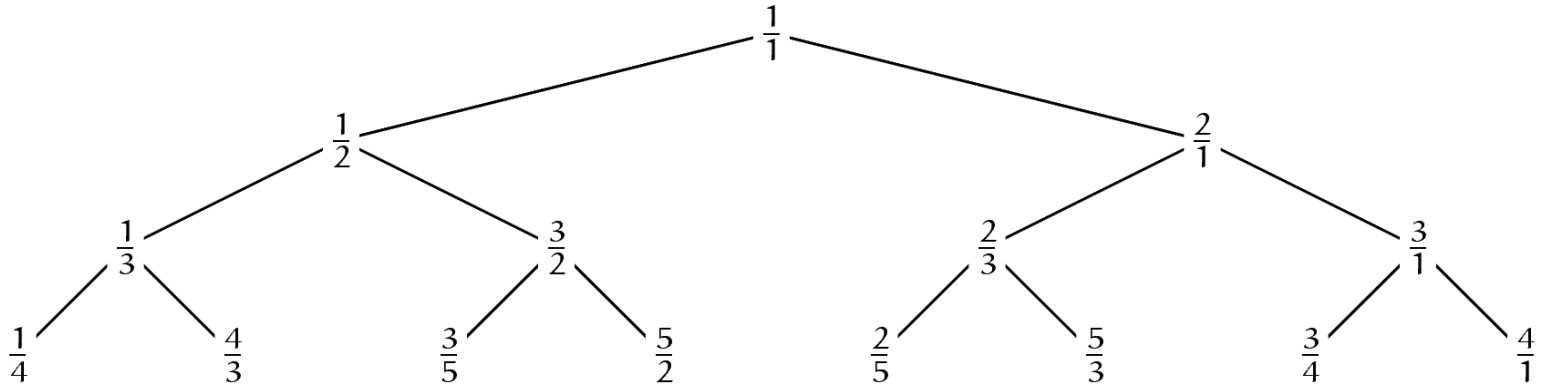
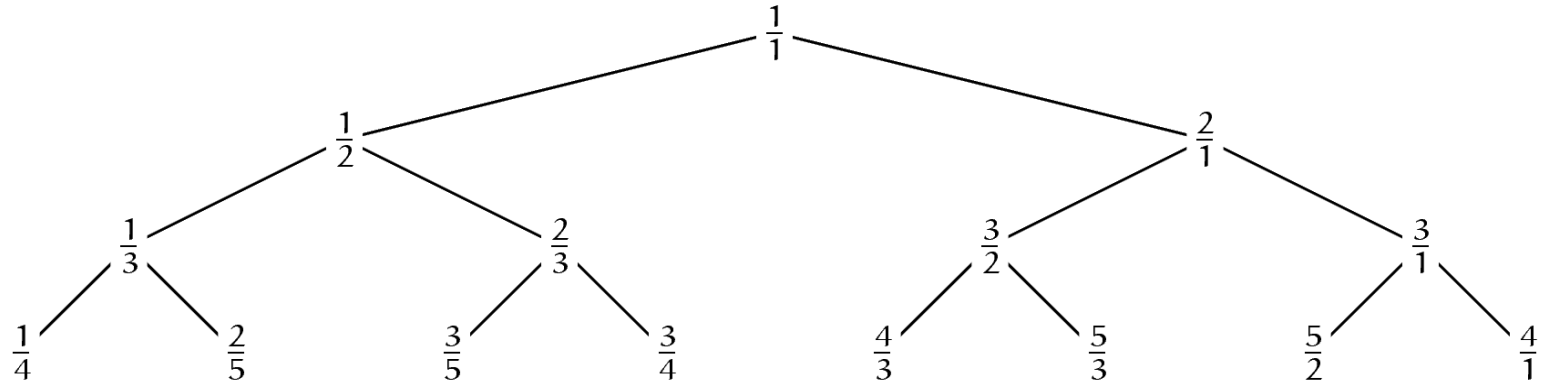
Enumerating the Rationals: Twice!

Roland Backhouse and João Ferreira*





* Funded by Fundação para a Ciência e a Tecnologia, Portugal

Positive Rationals

- A positive rational in "lowest-form" is an ordered pair of positive, coprime integers
- Every rational m/n has unique "lowest-form" representation:

$$\frac{m/(m \nabla n)}{n/(m \nabla n)}$$

where ∇ ("nabla") denotes the GCD.

Euclid's Algorithm

$\{\ 0 < m \ \wedge \ 0 < n\ \}$

$x, y := m, n\ ;$

$\{$ Invariant: $\ 0 < x \ \wedge \ 0 < y \ \wedge \ x \nabla y = m \nabla n\ \}$

do

$\quad x < y \longrightarrow \ y := y - x$

$\quad \square\ y < x \longrightarrow \ x := x - y$

od

$\{\ x = y = m \nabla n\ \}$

Extended Euclid's Algorithm

$\{\ 0 < m \ \wedge \ 0 < n\ \}$

$x, y := m, n \ ; \quad C := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

{ Invariant: $(x\ y) = (m\ n) \times C$

where $A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$ and $B = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$ }

do

$x < y \longrightarrow (x\ y) := (x\ y) \times A \ ; \quad C := C \times A$

$\square\ y < x \longrightarrow (x\ y) := (x\ y) \times B \ ; \quad C := C \times B$

od

$\{\ x = y = m \nabla n \ \wedge \ (x\ y) = (m\ n) \times C\ \}$

Extended Euclid's Algorithm

On termination, for arbitrary m and n:

$$\begin{pmatrix} m \vee n & m \vee n \end{pmatrix} = \begin{pmatrix} m & n \end{pmatrix} \times C$$

It follows that

$$\begin{pmatrix} 1 & 1 \end{pmatrix} \times C^{-1} = \begin{pmatrix} m/(m \vee n) & n/(m \vee n) \end{pmatrix}$$

Note

- m and n uniquely define C
- C^{-1} uniquely defines a rational m/n
  (all finite products of A^{-1} and B^{-1} are different)

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = A^{-1} \qquad L = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = B^{-1}$$

$$R^T = L$$

$\{ \; 0 < m \;\wedge\; 0 < n \; \}$

$x, y := m, n \;\; ; \;\; D := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

{ Invariant: $(x \;\; y) \times D = (m \;\; n)$ }

do

$x < y \longrightarrow (x \;\; y) := (x, y) \times A \;\; ; \;\; D := R \times D$

$\square \; y < x \longrightarrow (x \;\; y) := (x, y) \times B \;\; ; \;\; D := L \times D$

od

$\{ \; (1 \;\; 1) \times D = \left(\frac{m}{(m \circ n)} \qquad \frac{n}{(m \circ n)} \right) \; \}$

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = A^{-1} \qquad L = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = B^{-1}$$

$\{ \quad 0 < m \quad \wedge \quad 0 < n \quad \}$

$x, y := m, n \quad ; \quad E := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$\{$ Invariant: $\quad E \times \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} m \\ n \end{pmatrix} \quad \}$

do

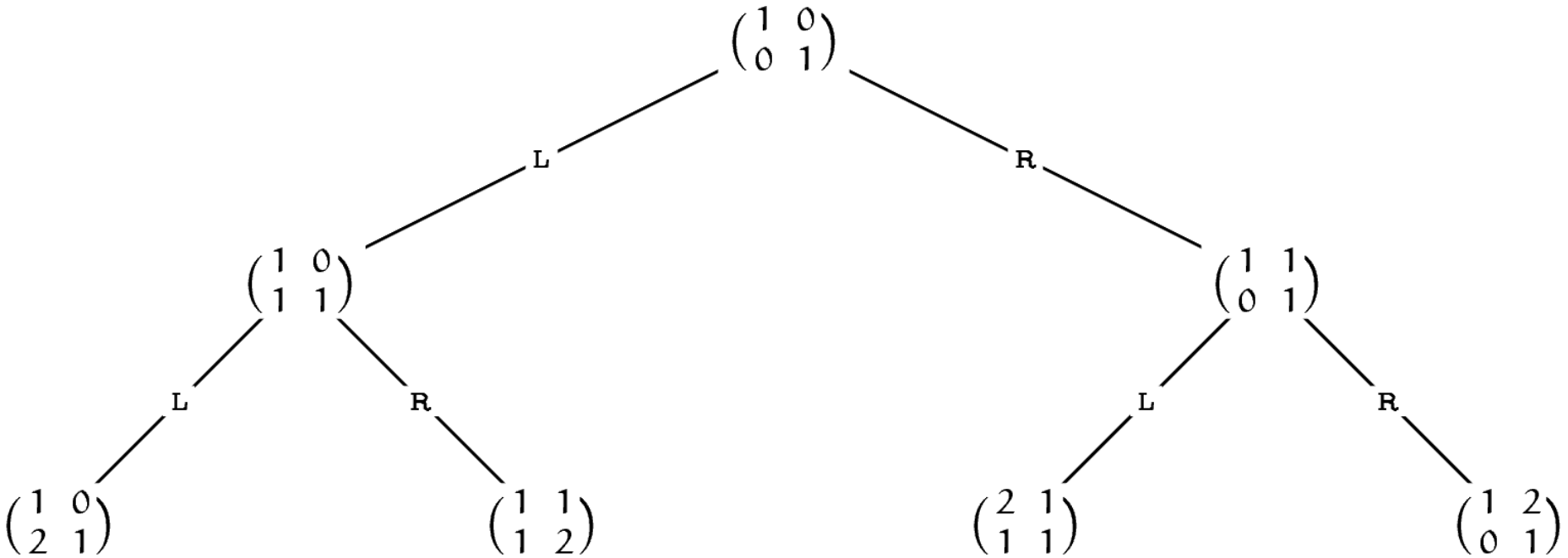
$\quad x < y \longrightarrow \quad \begin{pmatrix} x \\ y \end{pmatrix} := B \times \begin{pmatrix} x \\ y \end{pmatrix} \; ; \quad E := E \times L$

$\quad \square \; y < x \longrightarrow \quad \begin{pmatrix} x \\ y \end{pmatrix} := A \times \begin{pmatrix} x \\ y \end{pmatrix} \; ; \quad E := E \times R$

od

$\{ \quad E \times \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} m / (m \nabla n) \\ n / (m \nabla n) \end{pmatrix} \quad \}$

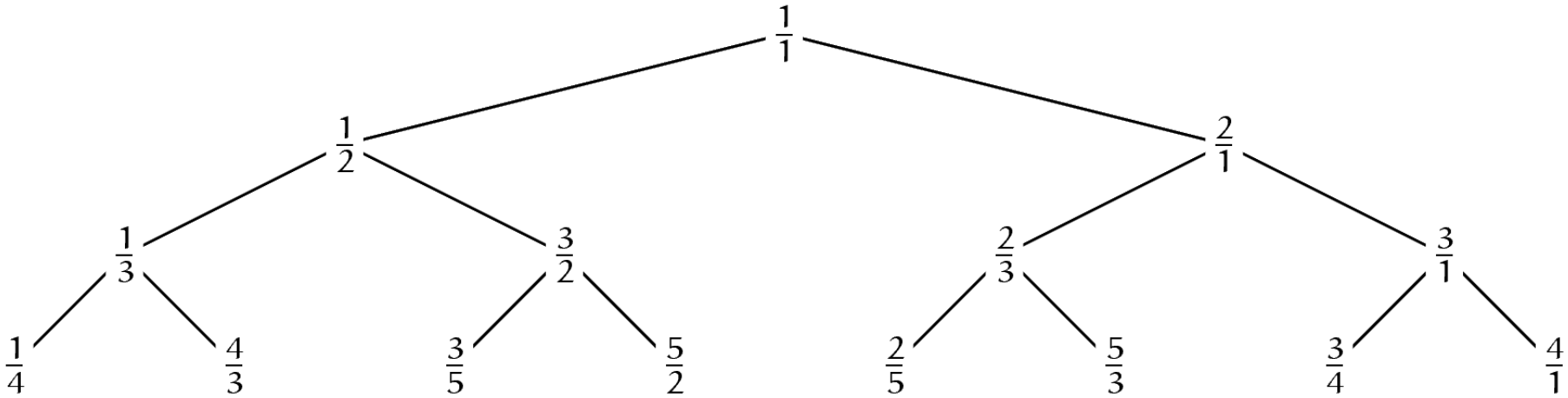
Tree of Matrices



$$L = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

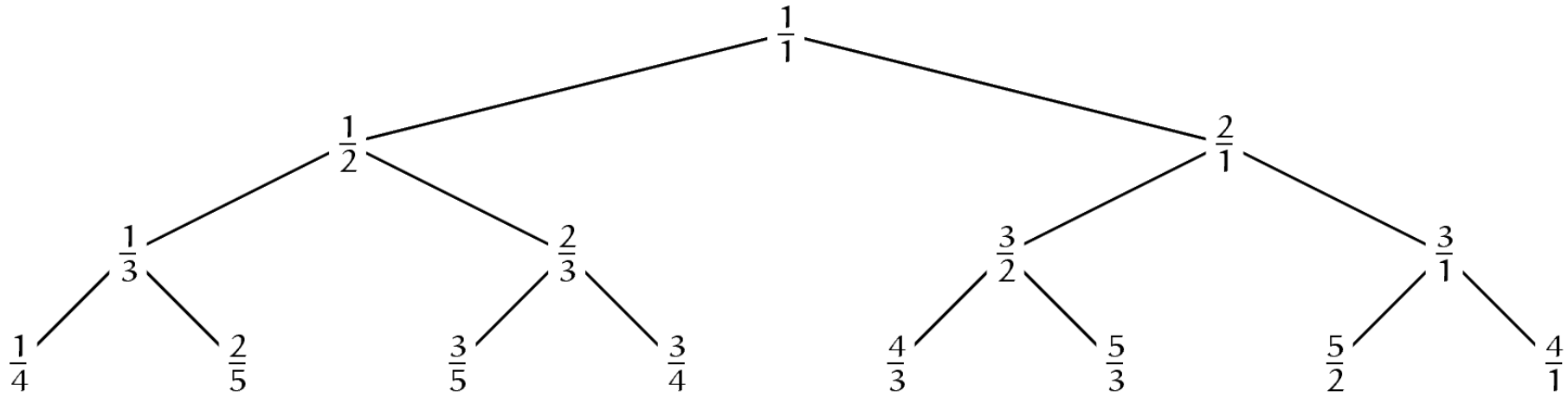
$$R = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

Calkin-Wilf Tree



Premultiply by $\begin{pmatrix} 1 & 1 \end{pmatrix}$

Stern-Brocot Tree



Postmultiply by $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$

Haskell Implementation (1)

```
type Entry  = Integer
type Column = [Entry]
type Matrix = [Column]

data Tree a              = Node (a, Tree a, Tree a)
foldt f (Node (a, l, r)) = f (a, foldt f l, foldt f r)
unfoldt f x              = let (a, y, z) = f x in Node (a, unfoldt f y, unfoldt f z)
```

Tree of matrices :

```
mTree :: Tree Matrix
mTree = unfoldt level matId
  where level m = (m, m × matL, m × matR)
```

Haskell Implementation (2)

Calkin-Wilf Tree:

```
cwTree :: Tree Rational
cwTree = foldt rat mTree
  where rat (m, l, r) = Node (mkCWRat ([[1], [1]] × m), l, r)

mkCWRat :: Matrix → Rational
mkCWRat [[m], [n]] = n/m
```

Stern-Brocot Tree:

```
sbTree :: Tree Rational
sbTree = foldt rat mTree
  where rat (m, l, r) = Node (mkSBRat (m × [[1, 1]]), l, r)

mkSBRat :: Matrix → Rational
mkSBRat [[m, n]] = m/n
```

Enumerating the Matrices

- There is a bijection between strings of "R"s and "L"s and products of Rs and Ls
- Finding the "next" matrix is equivalent to finding the next string in the lexicographic ordering
- Next string of the same length:

$$tLR^j \longrightarrow tRL^j$$

- Next matrix in the same level:

$$T \times L \times R^j \longrightarrow T \times R \times L^j$$

Computing the "next" matrix

$$(T \times L \times R^j) \times (R^{-j} \times L^{-1} \times R \times L^j) = T \times R \times L^j$$

where

$$R^{-j} \times L^{-1} \times R \times L^j = \begin{pmatrix} 2j+1 & 1 \\ -1 & 0 \end{pmatrix}$$

How to determine j?

do

$x < y \longrightarrow (x\ y) := (x, y) \times A \ ; \quad D := R \times D$

$\square\ y < x \longrightarrow (x\ y) := (x, y) \times B \ ; \quad D := L \times D$

od

Therefore, $j = \left\lfloor \frac{n-1}{m} \right\rfloor$

Computing the "next" matrix

$$j = \left\lfloor \frac{n-1}{m} \right\rfloor$$

On termination of the algorithm:

$$(1 \;\; 1) \times D = (m \;\; n)$$

That is, if

$$D = \begin{pmatrix} D_{00} & D_{01} \\ D_{10} & D_{11} \end{pmatrix}$$

then $m = D_{00} + D_{10}$ and $n = D_{01} + D_{11}$.

Therefore,

$$j = \left\lfloor \frac{n-1}{m} \right\rfloor = \left\lfloor \frac{D_{01} + D_{11} - 1}{D_{00} + D_{10}} \right\rfloor .$$

Enumerating the matrices

- D is a power of R exactly when the rationals are integers — easy to test!

$D := I$

do

$D_{00} + D_{10} = 1 \longrightarrow D := \begin{pmatrix} 1 & 0 \\ D_{01} + D_{11} & 1 \end{pmatrix}$

□

$D_{00} + D_{10} \neq 1 \longrightarrow j := \left\lfloor \frac{D_{01} + D_{11} - 1}{D_{00} + D_{10}} \right\rfloor \; ;$

$D := D \times \begin{pmatrix} 2j+1 & 1 \\ -1 & 0 \end{pmatrix}$

od

Haskell Implementation (3)

List of matrices:

$$
\begin{aligned}
&nextM :: Matrix \to Matrix \\
&nextM\ [[1,0],[n,1]] = [[1,n+1],[0,1]] \\
&nextM\ [c0, c1] = \mathbf{let}\ j = \lfloor ((sum\ c1) - 1)/(sum\ c0) \rfloor \\
&\qquad\qquad\qquad\qquad k = 2 \times j + 1 \\
&\qquad\qquad\qquad\qquad ck = map\ (k\times)\ c0 \\
&\qquad\qquad\qquad \mathbf{in}\ \ [zipWith\ (-)\ ck\ c1, c0]
\end{aligned}
$$

$$
\begin{aligned}
&mats :: [Matrix] \\
&mats = iterate\ nextM\ matId
\end{aligned}
$$

List of ~~Calkin-Wilf~~ Stern-Brocot rationals:

$$
\begin{aligned}
&cwEnum :: [Rational] \\
&cwEnum = map\ mkCW\ mats \\
&\quad \mathbf{where}\ mkCW = mkCWRat \circ (premul\ [[1],[1]]) \\
&premul\ v\ m = v \times m
\end{aligned}
$$

(Posmul [1,1])

posmul v m = m × v

Optimising Calkin-Wilf Enumeration

- The matrix $\begin{pmatrix} 2j+1 & 1 \\ -1 & 0 \end{pmatrix}$ is a function of $(1\ 1) \times D$

- Next matrix:

$$D := D \times J.((1\ 1) \times D)$$

- Next Calkin-Wilf rational:

$$(1\ 1) \times (D \times J.((1\ 1) \times D))$$

- Matrix multiplication is associative:

$$((1\ 1) \times D) \times J.((1\ 1) \times D)$$

Optimising Calkin-Wilf Enumeration

We get the following optimised algorithm:

$m, n := 1, 1$

do

$m = 1 \longrightarrow m, n := n+1, m$

□

$m \neq 1 \longrightarrow m, n := \left(2 \times \left\lfloor \frac{n-1}{m} \right\rfloor + 1\right) \times m - n, m$

od

(Note that the test for change in level is also a function of $(1\ 1) \times D$)

Optimising Calkin-Wilf Enumeration

- $\left\lfloor \frac{n-1}{m} \right\rfloor = \left\lfloor \frac{n}{m} \right\rfloor$ for $m \perp n$ and $m \neq 1$
- $\left(2 \times \left\lfloor \frac{n}{m} \right\rfloor + 1\right) \times m - n = n + 1$ when $m = 1$

Hence, we can eliminate the case analysis:

```
m, n := 1, 1
do
    m, n := (2 × ⌊n/m⌋) × m − n , m
od
```

(This is the algorithm discovered by Newman)

Optimising Calkin-Wilf Enumeration

In Haskell:

$$
\begin{aligned}
&cwnEnum :: [Rational] \\
&cwnEnum = iterate\ nextCW\ 1/1
\end{aligned}
$$

$$
\begin{aligned}
&nextCW :: Rational \to Rational \\
&nextCW\ r = \mathbf{let}\ (n, m) = (numerator\ r, denominator\ r) \\
&\qquad\qquad\quad j\ \ \ \ \ \ = \lfloor n/m \rfloor \\
&\qquad\qquad \mathbf{in}\ \ m/((2\times j + 1)\times m - n)
\end{aligned}
$$

Conclusions

- Our goal is to improve the effectiveness of algorithmic problem solving
- Matrices in Extended Euclid's Algorithm combine concision and precision
- Our construction makes "obvious" how to enumerate the rationals in Stern-Brocot order
- More importantly: we show the duality between both trees

Questions ?