BRAIDING IN CIRCLES

CHUNG-CHIEH SHAN AND DYLAN THURSTON

We use rope to make circular braids such as this: This design is the circular analogue of the *fishbone* hair braid. Thanks to the circularity, we could make it from a single strand of rope, which is typical. In the first half of our demo, we will teach everyone to make a simple circular braid from a single strand of rope. We will provide the rope.

We draw crossing diagrams to plan the construction of these braids. For example, the design above corresponds to the following diagram:

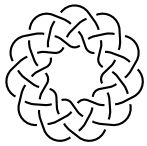
The diagram shows the same strand of rope multiple times. Braiding proceeds from left to right. The red curve follows how rope is newly laid down at each point in time, whereas black curves keep track of rope previously laid down, like trailing voices in a musical round.

We built a monadic combinator library in Haskell to describe braid designs in general and produce these diagrams automatically. Using this library, we can describe and diagram a braid—or a parameterized family of braids—by listing its crossings in sequence. For example, the simple circular braid that we will teach corresponds to the following diagram:

This diagram is produced by the code below (using the combinators *crossing*, *advance*, and *begin*):

```
diagonals = Stack ⟨√2, √2⟩ ⟨√2, −√2⟩
unit = replicateM_ 11 $ do crossing 2 ↘ ⟨−5, −5⟩ diagonals
                           crossing   1 ↘ ⟨0, 0⟩   diagonals
                           crossing     0 ↘ ⟨5, 5⟩ diagonals
                           advance ⟨10, 0⟩
main = putStr $ svg $ weave (begin 0 (red Open) ≫ unit ≫
                               replicateM_ 3 (begin 0 (paint Open) ≫ unit)) 4
```

The *circular* combinator, which has a rank-2 type, draws the completed circular braid:



```
main = putStr $ svg $ weave (begin 0 (paint Closed) ≫ circular unit) 4
```

In the second half of our demo, we will show off our combinator library and highlight its use of second-order functions, rank-2 types, and lazy evaluation. For example, the two-strand design below takes only 20 lines of code to describe.