1
Bridge Crossings

Problem

Four people begin on the same side of a bridge. You must send them across to the other side in the fastest time possible. It is night. There is one flashlight. A maximum of two people can cross at a time. Any party who crosses, either one or two people, must have the flashlight to see. The flashlight must be walked back and forth, it cannot be thrown, etc. Each person walks at a different speed. A pair must walk together at the rate of the slower person's pace, based on this information: Person 1 takes $t_1 = 1$ minutes to cross, and the other persons take $t_2 = 2$ minutes, $t_3 = 5$ minutes, and $t_4 = 10$ minutes to cross, respectively.

Günter Rote[1] gives a very elegant solution to this puzzle.

[1] Günter Rote. *Crossing the Bridge at Night*. World Wide Web, http://page.mi.fu-berlin.de/~rote/Papers/pdf/Crossing+the+bridge+at+night.pdf, 2002

How many ways are there to let n people cross the bridge under the rules of the original puzzle ?

There are $\binom{n}{2}$ ways to send the first pair over to the other side, there are 2 ways to send the flashlight back with somebody from that side. Now there are $\binom{n-1}{2}$ ways to send the next pair over to the other side from the remaining $n-1$ people on this side and then there are 3 ways to send the flashlight back with somebody from that side etc.

Using the basic product counting principle from combinatorics we get the number of ways P to let n people cross the bridge

$$\begin{aligned} P &= \binom{n}{2} 2 \binom{n-1}{2} 3 \binom{n-2}{2} 4 \dots (n-1) \binom{2}{2} \\ &= (n-1)! \prod_{k=0}^{n-2} \binom{n-k}{2} \end{aligned} \tag{1.1}$$

Taking the product from (1.1) and using the definition of a binomial coefficient we get:

$$\prod_{k=0}^{n-2}\binom{n-k}{2} = \prod_{k=0}^{n-2}\frac{(n-k)!}{2!(n-k-2)!} \tag{1.2}$$

With:

$$\begin{aligned} P_k &= \frac{(n-k)!}{2!(n-k-2)!} \quad \text{and} \\ p_k &= (n-k)! \end{aligned} \tag{1.3}$$

we get:

$$P_k = \frac{p_k}{2!p_{k+2}} \tag{1.4}$$

The product of these P_k can now be simplified to:

$$\begin{aligned} \prod_{k=0}^{n-2} P_k &= \prod_{k=0}^{n-2}\frac{(n-k)!}{2!(n-k-2)!} \\ &= \frac{1}{(2!)^{n-1}}\prod_{k=0}^{n-2}\frac{p_k}{p_{k+2}} \\ &= \frac{1}{2^{n-1}}\frac{p_0}{p_2}\frac{p_1}{p_3}\frac{p_2}{p_4}\cdots\frac{p_{n-3}}{p_{n-1}}\frac{p_{n-2}}{p_n} \\ &= \frac{1}{2^{n-1}}\frac{p_0p_1}{p_{n-1}p_n} \\ &= \frac{1}{2^{n-1}}n!(n-1)! \end{aligned} \tag{1.5}$$

Using (1.5) we get the solution

$$P = \frac{n!((n-1)!)^2}{2^{n-1}} \tag{1.6}$$

For four people this comes to an astonishing 108 ways to cross the bridge under the rules of the puzzle.

Generating the ways

This section shows a small Haskell program that generates all the possible ways to cross the bridge. It has a helper function *pairs* that generates a list of all possible pairs from a set. It then defines two mutually recursive functions *bridgecrossleft* and *bridgecrossright* for crossing the bridge from the left side as pairs and for a flashlight carrier coming back from the right. The functions pass along the states on the left bank *lbs* and the right bank *rbs*. They generate all possible crossings in their respective direction given the current state. For pairs crossing from the left tuples have the respective pair and for people coming back from the right tuples have the same person in both positions of the tuple. The functions collect the resulting combinations in a list of lists of tuples *rs* (Fig. 1.1). *bridgecross* is the main function taking a list

and calling *bridgecrossleft* because we start on the left with all possible ways of crossing of the first pair.

Calling *bridgecross [1, 2, 3]* we get this result:

```
[
  [(1,2),(1,1),(1,3)],
  [(1,2),(2,2),(2,3)],
  [(1,3),(1,1),(1,2)],
  [(1,3),(3,3),(3,2)],
  [(2,3),(2,2),(2,1)],
  [(2,3),(3,3),(3,1)]
]
```

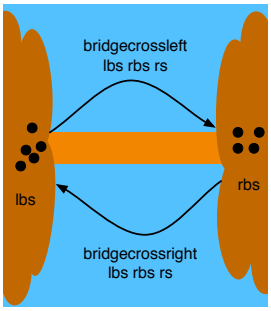

Figure 1.1: Two mutually recursive functions *bridgecrossleft* and *bridgecrossright*.

Listing 1.1: Haskell code

```
pairs :: [a] -> [(a, a)]

pairs xs = let
              rmap :: (a -> [a] -> [b]) -> [a] -> [b]

              rmap f (x:xs) = (f x xs) ++ (rmap f xs)

              rmap f [] = []

              mpairs :: (a -> [a] -> [(a, a)])

              mpairs x xs = map (\y -> (x, y)) xs

           in rmap mpairs xs

bridgecrossleft :: [Int] -> [Int] -> [(Int,Int)]
                                  -> [[(Int,Int)]]

bridgecrossleft lbs rbs rs
        = if (length lbs) >= 2 then
                let
                   ps = pairs lbs
                   f = (\(x,y) ->
                       (bridgecrossright
                          (filter (\z -> (z /= x)
                                      && (z /= y)) lbs)
                            (x:y:rbs) (rs ++ [(x, y)])))
                in foldl (++) [] (map f ps)
           else [rs]

bridgecrossright :: [Int] -> [Int] -> [(Int,Int)]
                                   -> [[(Int,Int)]]

bridgecrossright lbs rbs rs
         = if (length lbs) > 0 then
                 let
                    f = (\x ->
                         (bridgecrossleft (x:lbs)
                            (filter (\z -> (z /= x)) rbs)
                                (rs ++ [(x, x)])))
                 in foldl (++) [] (map f rbs)
            else [rs]

bridgecross :: [Int] -> [[(Int, Int)]]

bridgecross xs = bridgecrossleft xs [] []
```

Bibliography

Günter Rote. *Crossing the Bridge at Night*. World Wide Web, http://page.mi.fu-berlin.de/~rote/Papers/pdf/Crossing+the+bridge+at+night.pdf, 2002.