Exercise 2.43.

Louis Reasoner is having a terrible time doing exercise 2.42. His `queens` procedure seems to work, but it runs extremely slowly. (Louis never does manage to wait long enough for it to solve even the 6×6 case.) When Louis asks Eva Lu Ator for help, she points out that he has interchanged the order of the nested mappings in the `flatmap`, writing it as

```
(flatmap
 (lambda (new-row)
   (map (lambda (rest-of-queens)
          (adjoin-position new-row k rest-of-queens))
        (queen-cols (- k 1))))
 (enumerate-interval 1 board-size))
```

Explain why this interchange makes the program run slowly. Estimate how long it will take Louis's program to solve the eight-queens puzzle, assuming that the program in exercise 2.42 solves the puzzle in time T.

Answer.

Suppose the board size involved is n, that is, the problem is a size of n. We begin by compare the behavior of these two implementations.

In the original version, the nested mapping in the `flatmap` is

```
(flatmap
 (lambda (rest-of-queens)
   (map (lambda (new-row)
          (adjoin-position new-row k rest-of-queens))
        (enumerate-interval 1 board-size)))
 (queen-cols (- k 1)))
```

For each of the solutions to the problem of $k-1$ queens, `queen-cols` places an additional queen in each row of the kth column. Yet following this practice, `queen-cols` successively reduces the problem to one size smaller in a call to itself. Thus, `queen-cols` would call itself for $n+1$ times to generate all the solutions to place n queens. So the order of growth of the steps in this process is $\Theta(n)$.

In Louis's implementation, `queen-cols` generates all the solutions to place $k-1$ queens for each row of the kth column. It then adjoins the kth queen to each of these structures and filters them to obtain the result. Since all the solutions to place $k-1$ queens are identical, `queen-cols` computes them repeatedly for k times while placing queens in the kth column. Further more, this repetition occurs in all the reduced sizes of problem where `queen-cols` proceeds to place queens in the $(k-1)$th, $(k-2)$th, ..., 1st column. Hence, in every column to place a queen, the process generated by Louis's procedure is N times to that of the original `queens` procedure. Hence, it will takes up approximately $O(n^n)$ steps for Louis's `queens` procedure to produce the solution for n queens.

This enormous order of growth dramatically but strongly illustrates why Louis's implementation has such a bad performance. We now can assert that it will takes Louis's program approximately a time of $n^n \times T$ to solve the eight-queens puzzle.

Email address: informlarry@gmail.com