Exercise 3.30.

Figure 1 shows a *ripple-carry adder* formed by stringing together n full-adders. This is the simplest form of parallel adder for adding two n-bit binary numbers. The input A_1, A_2, A_3, ..., A_n and B_1, B_2, B_3, ..., B_n are the two binary numbers to be added (each A_k and B_k is a 0 or a 1). The circuits generates $S_1, S_2, S_3, ..., S_n$, the n bits of the sum, and C, the carry from the addition. Write a procedure `ripple-carry-adder` that generates this circuit. The procedure should take as arguments three lists of n wires each—the A_k, the B_k, and the S_k—and also another wire C. The major drawback of the ripple-carry adder is the need to wait for the carry signals to propagate. What is the delay needed to obtain the complete output from an n-bit ripple-carry adder, expressed in terms of the delays for and-gates, or-gates, and inverters?

Answer.

An n-bit ripple-carry adder can be degenerated into a `fuller-adder` of the first bits together with the ripple-carry adder of the rest of the lists. The simplest case lies where the lists are empty, in which the adder just terminates:

```
(define (ripple-carry-adder Ak Bk Sk C)
  (if (and (null? Ak) (null? Bk) (null? Sk))
      'ok
      (let ((A1 (car Ak))
            (B1 (car Bk))
            (S1 (car Sk))
            (C1 (make-wire))
            (remain-Ak (cdr Ak))
            (remain-Bk (cdr Bk))
            (remain-Sk (cdr Sk)))
        (set-signal! C1 (get-signal C))
        (full-adder A1 B1 C1 S1 C)
        (ripple-carry-adder remain-Ak remain-Bk remain-Sk C))))
```

Since a typical n-bit ripple-carry adder consists of n `full-adder`s, hence the delay time of it equals to n times the delay time of a `full-adder`:

$$\texttt{ripple-carry-adder-delay} = n \times \texttt{full-adder-delay} \tag{1}$$

On the other hand, a `full-adder` is itself made up of two `half-adder` and an or-gate, as figure 3.26 shows:

$$\texttt{full-adder-delay} = 2 \times \texttt{half-adder-delay} + \texttt{or-gate-delay} \tag{2}$$

We've also seen in figure 3.25 that, a typical `half-adder`, is constructed by two and-gates, one or-gates as well as an inverter. Notice that the first or-gate and the first and-gate run parallel to each other, in composing a `half-adder` and this reveals the way to computing the delay time of a `half-adder`:

$$\begin{aligned}\texttt{half-adder-delay} \;=\; & \max(\texttt{or-gate-delay}, \texttt{and-gate-delay}) + \texttt{or-gate-delay} \\ & + \texttt{and-gate-delay}\end{aligned} \tag{3}$$

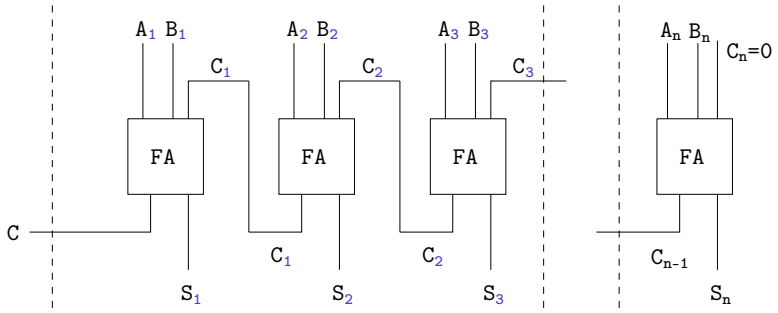

Figure 1. A ripple-carry adder for n-bit numbers.

Synthesize formula (1) to (3), we obtain the delay time of an n-bit ripple-carry adder:

$$\begin{aligned} \texttt{ripple-carry-adder-delay} \;=\; & 2\,n \times \max(\texttt{or-gate-delay}, \texttt{and-gate-delay}) \\ & +3\,n \times \texttt{or-gate-delay} + 2\,n \times \texttt{and-gate-delay} \end{aligned} \tag{4}$$