Problem

Let A be a regular language over {0,1}. Show that A has size–depth complexity (O(n),O(log n)).

Step-by-step solution

Step 1 of 1

Suppose a **language A** is defined in $\{0,1\}$ that consist every strings with an odd number of one's. The parity function computation can be used to test membership in A. The standard AND, OR and NOT operations can be used **to implement the two parity gate** $a \oplus b$ as $(a \wedge \neg b) \vee (\neg a \wedge b)$.

• Suppose $a_1, a_2, \dots, a_n$ be taken as the input to the circuit. There are many ways define a circuit with $O(n)$ size. One way to get a circuit for parity function is to construct gates g_i whereby $g_1 = a_1$ and $g_i = a_i \oplus g_{i-1}$ for $i \le n$. This construction uses $O(n)$ size and depth.

• Another way to do this, by **building a binary tree of gates** that computes the XOR function, where the XOR function is the same as the parity function and then implements **each XOR gate with two NOTs, two ANDs and one OR gates**. This construction uses $O(n)$ and $O(\log n)$ depth.

This construction is a significant improvement because it **uses exponentially less parallel time** than does the preceding construction. **Thus, the size-depth complexity of A is** $(O(n), O(\log n))$.

Comment