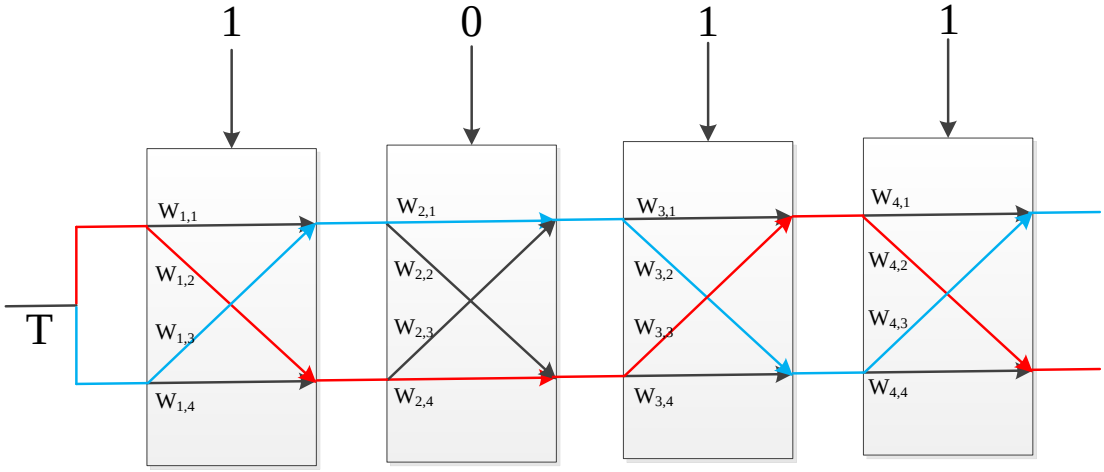
三种 Arbiter PUF 逻辑回归建模

有人问我要三种建模方式，应大家要求。



1. 4X64 参数建模

对 Arbiter PUF 的所有延迟段进行设参数。图中的$(w_{11}, w_{12}, w_{13}, w_{14})$对应之前图中的(p, s, t, q)，这时我们产生响应相当于比较(w12+w24+w33+w42)和(w13+w21+w32+w43)的大小。即判断以下两个矩阵对应位置相乘后求和的正负，

$$C = \begin{matrix} 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 1 \\ -1 & 0 & 1 & -1 \\ 0 & 1 & 0 & 0 \end{matrix} \qquad W = \begin{matrix} W_{11} & W_{21} & W_{31} & W_{41} \\ W_{12} & W_{22} & W_{32} & W_{42} \\ W_{13} & W_{23} & W_{33} & W_{43} \\ W_{14} & W_{24} & W_{34} & W_{44} \end{matrix}$$

可见激励(1011)和矩阵 C 是一一对应的，我们只需要对激励进行扩展就能完成建模。当然还有其他的设参数方式以及对应的激励扩展方式，但是万变不离其宗，建模还是要符合 Arbiter PUF 的工作原理，才能建出好的模型，大家加油。

2. 2X64 参数建模

这种建模方式很多组都做出来了，就是分别将平行路径和交叉路径的延迟差设参进行建模。设 a_i 为每段平行路径的延迟差（上减下），b_i 为交叉路径的延迟差（上减下），因此上图对应的两矩阵为：

$$C = \begin{matrix} 0 & -1 & 0 & 0 \\ 1 & 0 & -1 & 1 \end{matrix} \qquad W = \begin{matrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \end{matrix}$$

3. 64 参数建模

这是大家最困惑的建模方式了，也正因为这个建模可以做到，才体现了机器学习的强大。事实上，这种建模方式，学长原先也没试过，今天才做了下。讲讲思路吧，既然我们能对平行路径和交叉路径的延迟差设参，那么我们能不能将这两个参数相关起来，即使得 a + b = c，c 是多少？不需要知道，让机器学习自己去学吧，既然我们不要知道 c 是多少，那我们就把它当做 0，哈哈，这样就有 a + b = 0，即 b = -a 了。那么上图两矩阵为：

C = -1 -1 1 -1　　W = a_1 a_2 a_3 a_4

代码给你们了，三种建模方式都在里面，有兴趣的自己看下。