Triads and Dyads

Not dryads and naiads...

Charles Peirce, who is regarded among the greatest logicians and philosophers, wrote a paper in 1897 titled "The Logic of Relatives." Now we might want a guide to the logic of our own relatives, and in fact the word "brother" furnished Peirce's first example. But what Peirce meant by "relative" is a term that attains its meaning only in conjunction with another noun as object, *plus* a word or other sign giving the relation. Examples, the first two his, are "brother of Napoleon," "slayer of giants," and "among the greatest logicians and philosophers." The last has a wordless sign after "among" where the others have "of," but then the rest may seem to make it a four-way relation. However, Peirce gave a general calculus of how all four and higher-way relations in semiotics could be decomposed into twos and threes. Such "relatives" and other three-way relations, however, he proved to be irreducible within his system.

Today we consider an open problem about decomposing 3-ary finite automata.

Peirce's name is usually written with at least the 'S.' of his middle name. He was born Charles Sanders Peirce, but late in life he inserted "Santiago" before or replacing his middle name. As "St. James" in Spanish, this was said to honor his friend the philosopher William James, whose son Pierce designated as second heir after his own wife Juliette. However, earlier usages may have honored the European origins of Juliette herself. His taking up with Juliette before his divorce from his first wife became final was made a scandal that cost Peirce the one academic position he ever had, from Johns Hopkins University in 1884.

I was prompted to write this by a response from Jon Awbrey, who frequents our pages and writes the blog "Inquiry Into Inquiry." In Boolean logic every N-way function can be decomposed as a circuit of binary functions, indeed composed entirely of the binary NOR function, as Peirce himself discovered. Awbrey termed it a "trap" to infer that this carries over to logical semantics. The rest of this post isn't related to Pierce, but we offer it toward the general question, when do threes decompose into twos?

Multi-tape Finite Automata

The familiar idea of a finite automaton M can be generalized to any number k of input tapes. If each input head must advance one cell on each move, then the input strings may as well be padded to the same length n with an extra symbol \$ added to the alphabet Σ. Then they can be shuffled into one string of length n over the alphabet $\Sigma' = \Sigma^k$, whereupon M becomes equivalent to the resulting ordinary one-tape finite automaton M'.

Things are more interesting when one or more heads are allowed to stay stationary on each move. Then the string-matching relation can be recognized by a *nondeterministic* 2-tape finite automaton (2-NFA). This relation consists of all pairs (x, y) such that y can be written as $y = uxv$ for some $u, v \in \Sigma^*$. The 2-NFA pauses its first tape head and advances the second until it guesses where x may start in the input y, and accepts if and when it matches all characters of x on consiecutive steps. It is easy to show that no 2-DFA can recognize this relation. Thus the equality "NFA = DFA" does not carry over to multiple tapes when pausing is allowed.

A three-way relation that is deterministically recognizable is the concatenation relation $C(x, y, z) = (xy = z)$. With x and y on two separate tapes, a 3-DFA can recognize where x stops upon reading the (implicit or explicit) terminal \$ on that tape, and then begin matching the rest of the third tape against y. In an early paper, I showed that there is a bijection $f : \{0,1\}^* \times \{0,1\}^* \longrightarrow \{0,1\}^*$ whose graph is similarly recognizable as a three-way relation. Indeed I gave a two-tape finite *transducer* to compute it. This showed a pairing function that is linear-time computable and invertible, while previous examples used integer multiplication for which linear time is unknown.

Multi-tape Finite Transducers and the Problem

The definition of a k-tape finite-state transducer (k-FST) is straightforward: Each transition has the form

$$(p, (c_1, \ldots, c_k), (D_1, \ldots, D_k), y, q)$$

where p and q are states, $c_1, \ldots, c_k \in \Sigma$ are input characters or \$, each D_i is 'S' for "stay" or 'R' for move one cell right, and $y \in \Sigma^*$. To make the computation straightforward, we can stipulate that at least one D_i in each instruction is an R, and that the machine goes to a designated halting state only when each $c_i =$ \$. Then on

inputs $x = (x_1, \ldots, x_k)$ where $\max |x_i| = n$, the machine runs for at most $k(n+1)$ steps.

An example of a function computed by a 3-FST is the ternary concatenation function $c_3(x, y, z) = xyz$. The latter two tapes pause while x is copied to the output, then y, then z. This function, however, can be written as the composition

$$c_3(x, y, z) = c_2(c_2(x, y), z)$$

of the usual binary concatenation function. This leads straight into our question:

> Can every 3-FST function be computed by a finite composition of 2-FSTs?

We can think of other problems involving relations and functions computed by binary and ternary finite automata that seem to have ready answers. One is whether every 3-DFA relation can be written as a Boolean combination of 2-DFA relations on its three pairs of inputs—consider the relation $z = x \oplus y$ where $\oplus$ is bitwise exclusive-or. The answer for the function question has, however, eluded me—I suspect it is *no*, but have not even found a "killer" candidate function for being 3-FST only.

Open Problems

Can you prove the answer?

What about other cases of k-way into 2-way decomposition?