Enumerating cyclical decompositions with Stirling numbers

devblogs.microsoft.com/oldnewthing/20141006-00

October 6, 2014

Raymond Chen

This whole enumeration nightmare-miniseries started off with Stirling numbers of the second kind. But what about Stirling numbers of the first kind? Those things ain't gonna enumerate themselves!

The traditional formulation of the recursion for Stirling numbers of the first kind (unsigned version, since it's hard to enumerate negative numbers) goes like this:

$$c(n+1, k) = n \cdot c(n, k) + c(n, k-1).$$

although it is more convenient from a programming standpoint to rewrite it as

$$c(n, k) = (n-1) \cdot c(n-1, k) + c(n-1, k-1).$$

The Wikipedia article explains the combinatorial interpretation, which is what we will use to enumerate all the possibilities.

- The first term says that we recursively generate the ways of decomposing $n-1$ items into k cycles, then insert element n in one of $n-1$ ways.
- The second term says that we recursively generate the ways of decomposing $n-1$ items into $k-1$ cycles, then add a singleton cycle of just n.

```
function Stirling1(n, k, f) {
 if (n == 0 && k == 0) { f([]); return; }
 if (n == 0 || k == 0) { return; }
 // second term
 Stirling1(n-1, k-1, function(s) { f(s.concat([[n]])); });
 // first term
 Stirling1(n-1, k, function(s) {
  for (var i = 0; i 0; j  ) {
    f(s.map(function(c, index) {
     return i == index ? c.slice(0, j).concat(n, c.slice(j)) : c;
    }));
   }
  }
 });
}
Stirling1(5, 3, function(s) { console.log(JSON.stringify(s)); });
```

The inner loop could just as easily gone

```
   for (var j = 0; j < s[i].length; j++) {
```

but I changed the loop control for style points. (It makes the output prettier.)

Raymond Chen

Follow