Letters to the Editor

The Quarts or Two Pail or Pitcher pouring problem revisited ..Again

Dear Editor:

The pitcher pouring problem posed by Ihor in the Clime News (V. 2 N. 1) is addressed in Volume 2 of *Computer Science Logo Style* (MIT Press, 1986). My program is based on the ideas of a tree of possible states (i.e., amounts of water in each pitcher after a sequence of pourings) and it finds an optimal solution by a breadth-first search of this tree. Someone might find it interesting to combine my solution procedures with Ihor's graphical representation procedures. Of course, if you want to use the problem to develop creative problem solving in human beings, a computer solution would be unhelpful. But the program can lead to some interesting insights into the mathematics of the problem. For example, with two pitchers there are three possible sources and destinations for each pouring step; the third is the river. That means that each at level n of the solution tree gives rise to six states at level n+1 (three sources times two destinations for each, since it doesn't make sense to pour from the pitcher into itself). So, in principle, the breadth of the tree grows exponentially as the number of steps in the solution increases. However, it turns out that almost all possible outpourings are redundant, in the sense that they result in a state that you've already seen higher in the tree. For the configuration Ihor discusses, a 4-quart pitcher and a 9-quart pitcher, the number of new states found at each level of the tree is always 2 except in the second step, where it reaches 3. With these pitchers, there are 10 possible goal quantities (0 through 9 quarts); by letting the program print out the new states found at each level, we can see that the hardest of these goals are 2 quarts and 7 quarts, each requiring ten pouring steps.

Brian Harvey
2634 Virginia St.
Berkley, CA 94709

• • • • • • • • •

Dear Editor,

I really liked the two pail problem. Here's a start of a solution: Your remark that some choices of pail sizes and targets won't work is the key. If the pail sizes are A and B, then a target T won't have a chance unless you can find integers F and S so that: FA + SB = T. Furthermore, T can't be bigger than A + B. It turns out that these are the only obstacles; if T is between 0 and A + B and T is a combination of A and B (T = FA + SB), then you can juggle the pails to produce T. Rather than a proof, let me give you an example: suppose that A = 30, B = 42 and T = 48 (I'd have to exercise for a few weeks before I could handle that much water). I proceed as follows:

1. 48 is 18 mod 30, so I only need to build 18. That's because if I get 18, I can put it in the big pail and "spice" with a whole little pail.

2. 18=9*30 - 6*42. These may not be the "minimal" coefficients that work, but they work. So, I want to fill the little pail 9 times and empty the big pail 6 times to get 18. Here are the steps, showing the contents of each pail at each step:

Step	A	B	Step	A	B
1	0	0	16**	24	0
2*	30	0	17	0	24
3	0	30	18*	30	24
4*	30	30	19	12	42
5	18	42	20**	12	0
6**	18	0	21	0	12
7	0	18	22*	30	12
8*	30	18	23	0	42
9	6	42	24**	0	0
10**	6	0	25*	30	0
11	0	6	26	0	30
12*	30	6	27*	30	30
13	0	36	28	18	42
14*	30	36	29**	18	0
15	24	42	30	0	18

The * steps show where a little pail is filled and the ** shows when you dump a 42. Notice that this is really the long way around the block; we had 18 in our lap (well, in our bucket) at step 6.

So the question comes down to this: For what numbers T between 0 and A (because we reduce mod A first) can we write T = FA + SB? There is an important theorem from number theory (the motivation for the definition of PID's) that says that the numbers of the form FA + SB are precisely the multiples of GCD (A,B). Of course, we can model GCD in Logo via Euclid's algorithm. The wonderful thing is that we can also work the algorithm backwards to get a Logo model that calculates the first coefficient F and the second coefficient S so that FA + SB = GCD (A,B). Then, if T is a multiple of GCD (A,B), you can multiply both sides of this equation by T/GCD(A,B) to obtain an expression for T of the form T = (something)A + (something)B. The rest (probably the hard part) is to make the Logo functions say something that people would recognize (drawing the pails, etc). Anyway, here's a sketch of some functions: First, Euclid:

```
TO GCD :B :A
IF (MOD :B :A) = 0 [OP :A]
OP GCD :A (MOD :B :A)
END
```

*See **Quarts, Pail, or Pitcher**...page 18*