Proving the Correctness of a Program for Computing GCD and LCM

Pritom Rajkhowa

May 2019

1 Motivating Examples

1.1 Method 1

```
/pre( a>0 && b>0 )/
{
    x=a;
    y=b;
    u=a;
    v=b;
    while(x!=y) {
          if (x>y) {
                 x=x-y;
                 u=u+v;
            }
            else
            {
                y=y-x;
                v=u+v ;
            }
     }
     result=(u+v)/2;
}
/post( x=GCD(a,b) && result == LCM(a,b) )/
```

Using Hoare's logic, this can be proved using the following loop invariant:

$$GCD(x,y) = GCD(a,b) \wedge x > 0 \wedge y > 0 \wedge x \times v + y \times u == 2 \times a \times b.$$

Let's see how we prove this using the translation from [1] and extended in [2] and [3].

$$\begin{aligned}
&x_5(0) = a,\\
&y_5(0) = b,\\
&u_5(0) = a,\\
&v_5(0) = b,\\
&x_5(n_1+1) = ite(x_5(n_1) > y_5(n_1), x_5(n_1) - y_5(n_1), x_5(n_1)),\\
&u_5(n_1+1) = ite(x_5(n_1) > y_5(n_1), u_5(n_1) + v_5(n_1), u_5(n_1)),\\
&y_5(n_1+1) = ite(x_5(n_1) > y_5(n_1), y_5(n_1), y_5(n_1) - x_5(n_1)),\\
&v_5(n_1+1) = ite(x_5(n_1) > y_5(n_1), v_5(n_1), u_5(n_1) + v_5(n_1)),\\
&x_5(N_1) = y_5(N_1),\\
&\forall n_1.n_1 < N_1 \rightarrow x_5(n_1) \neq y_5(n_1).\\
&x_1 = x_5(N_1),\\
&y_1 = y_5(N_1),\\
&u_1 = u_5(N_1),\\
&v_1 = v_5(N_1),\\
&result_1 = (u_5(N_1) + v_5(N_1))/2.
\end{aligned}$$

These formulas use conditionals. We can write them as ordinary functions using boolean expressions: for any formula φ, we let $\varphi = 1$ if φ is true and $\varphi = 0$ if φ is false. Notice further that all of the functions like $x_5(n)$ and N are dependent on a and b. We can make this explicit. We thus get the following formulas:

$x_5(a,b,0) = a,$
$y_5(a,b,0) = b,$
$u_5(a,b,0) = a,$
$v_5(a,b,0) = b,$
$x_5(a,b,n_1+1) = ite(x_5(a,b,n_1) > y_5(a,b,n_1), x_5(a,b,n_1) - y_5(a,b,n_1), x_5(a,b,n_1)),$
$u_5(a,b,n_1+1) = ite(x_5(a,b,n_1) > y_5(a,b,n_1), u_5(a,b,n_1) + v_5(a,b,n_1), u_5(a,b,n_1)),$
$y_5(a,b,n_1+1) = ite(x_5(a,b,n_1) > y_5(a,b,n_1), y_5(a,b,n_1), y_5(a,b,n_1) - x_5(a,b,n_1)),$
$v_5(a,b,n_1+1) = ite(x_5(a,b,n_1) > y_5(a,b,n_1), v_5(a,b,n_1), u_5(a,b,n_1) + v_5(a,b,n_1)),$
$x_5(a,b,N_1) = y_5(a,b,N_1),$
$\forall n_1.n_1 < N_1 \rightarrow x_5(a,b,n_1) \neq y_5(a,b,n_1).$
$x_1(a,b) = x_5(a,b,N_1),$
$y_1(a,b) = y_5(a,b,N_1),$
$u_1(a,b) = u_5(a,b,N_1),$
$v_1(a,b) = v_5(a,b,N_1),$
$result_1(a,b) = (u_5(a,b,N_1) + v_5(a,b,N_1))/2.$

Now let T be the set of the above axioms, with a, b, n_1 universally quantified over natural numbers. As far as program semantics is concerned, once we have obtained this theory T, we are done. The rest depends on the particular domain that the program is about. For example, the assertions about the program become the following theorems to prove:

$$T \models \forall a, b.a > 0 \wedge b > 0 \rightarrow x_1(a,b) = GCD(a,b) \wedge result_1(a,b) = LCM(a,b). \quad (1)$$

Of course, the entailment " $\models$ " is understood to be under a number theory that includes axioms about LCM and GCD, and is undecidable. So it is unreasonable to expect a fully automated system will always be able to prove theorems like this. We assume the following theorems (definitions) about LCM and GCD:

$LCM(a,b) = ab/GCD(a,b),$
$GCD(a,a) = a,$
$a > b \rightarrow GCD(a,b) = GCD(a-b,b),$
$b > a \rightarrow GCD(a,b) = GCD(a,b-a).$

We prove 1 by cases as well. We first prove some properties about x, y, u and v.

An obvious one is that if $a > b$, then $x_5(a, b, 1) = a - b = x_5(a - b, b, 0)$ and $y_5(a, b, 1) = b = y_5(a - b, b, 0)$. Thus we have

$$a > b \rightarrow [x_5(a - b, b, n_1) = x_5(a, b, n_1 + 1) \wedge \\ y_5(a - b, b, n_1) = y_5(a, b, n_1 + 1)]. \quad (2)$$

From this, we conclude

$$a > b \rightarrow N_1(a, b) = N_1(a - b, b) + 1. \quad (3)$$

Thus if $a > b$, then

$$x_1(a, b) = x_5(a, b, N_1(a, b)) = x_5(a, b, N_1(a - b, b) + 1) \\ = x_5(a - b, b, N_1(a - b, b)) = x_1(a - b, b).$$

Similarly it can be seen that $x_1(a, a) = a$ and $b > a \rightarrow x_1(a, b) = x_1(a, b - a)$. Thus $x_1(a, b) = GCD(a, b)$.

For LCM, we show $result_1(a, b) = ab/GCD(a, b)$, that is

$$u_5(a, b, N_1(a, b)) + v_5(a, b, N_1(a, b)) = 2 \times a \times b/GCD(a, b). \quad (4)$$

We prove this by induction on $a + b$. The base case is $a + b = 2$, which implies that $a = b = 1$, and the result is obvious. Suppose the result holds for any $a + b < k$, we show it holds for $a + b = k$. If $a = b$, then the result holds. We assume $a > b$. The other case for $b > a$ is symmetric. Since $(a - b) + b < a + b$, by inductive assumption,

$$u_5(a-b, b, N_1(a-b, b))+v_5(a-b, b, N_1(a-b, b)) = 2\times(a-b)\times b/GCD(a-b, b).$$

Since $GCD(a-b, b) = GCD(a, b)$, we prove (4) by showing the following:

$$(a - b) \times (u_5(a, b, N(a, b)) + v_5(a, b, N(a, b))) \\ = a \times (u_5(a - b, b, N_5(a - b, b)) + v_5(a - b, b, N_5(a - b, b))). \quad (5)$$

The key to prove this is to observe that for $u + v$ to be divided by both a and b, there must be some k_1 and k_2 such that $u + v = k_1 \times a = k_2 \times b$. From the recurrence, we know that $u_5(a, b, n_1) + v_5(a, b, n_1)$ will have the form $m_1 \times a + m_2 \times b$. So for $u + v = k_1 \times a = k_2 \times b$ to be always true

for all a and b, it must be that $m_1 \times a = m_2 \times b$ when $n_1 = N_1(a, b)$. The condition at $N_1(a, b)$ is that $x = y$, and $x_5(a, b, n_1)$ and $y_5(a, b, n)$ will have the form $n_1 \times a - n_2 \times b$ and $n_3 \times b - n_4 \times a$, respectively. When they are equal at $N(a, b)$, we have $(n_1 + n_4) \times a = (n_2 + n_3) \times b$, which suggest that $m_1 = n_1 + n_4$ and $m_2 = n_2 + n_3$. How to do this automatically is of course an interesting question. In fact, there were work on using recurrences like what we have here for automatically discovering loop invariants for Hoare logic [1].

We rewrite x, y, u, v as

$$
\begin{aligned}
x_5(a, b, n_1) &= f_1(a, b, n_1) \times x_5(a, b, 0) - f_2(a, b, n_1) \times y_5(a, b, 0), \\
y_5(a, b, n_1) &= f_3(a, b, n_1) \times y_5(a, b, 0) - f_4(a, b, n_1) \times x_5(a, b, 0), \\
u_5(a, b, n_1) &= f_5(a, b, n_1) \times u5(a, b, 0) + f_6(a, b, n_1) \times v_5(a, b, 0), \\
v_5(a, b, n_1) &= f_7(a, b, n_1) \times u5(a, b, 0) + f_8(a, b, n_1) \times v_5(a, b, 0),
\end{aligned}
$$

and use $x_5(a, b, n_1 + 1)$ and other recurrences to get the following recurrences for $f_1 - f_8$:

$$
\begin{aligned}
& f_1(a, b, 0) = f_3(a, b, n_1) = 1, f_2(a, b, 0) = f_4(a, b, n_1) = 0, \\
& f_1(a, b, n_1 + 1) = f_1(a, b, n_1) + (x_5(a, b, n_1) > y_5(a, b, n_1)) \times f_4(a, b, n_1), \\
& f_2(a, b, n_1 + 1) = f_2(a, b, n_1) + (x_5(a, b, n_1) > y_5(a, b, n_1)) \times f_3(a, b, n_1), \\
& f_3(a, b, n_1 + 1) = f_3(a, b, n_1) + (\neg x_5(a, b, n_1) > y_5(a, b, n_1)) \times f_2(a, b, n_1), \\
& f_4(a, b, n_1 + 1) = f_4(a, b, n_1) + (\neg x_5(a, b, n_1) > y_5(a, b, n_1)) \times f_1(a, b, n_1), \\
& f_5(a, b, 0) = f_8(a, b, 0) = 1, f_6(a, b, 0) = f_7(a, b, 0) = 0, \\
& f_5(a, b, n_1 + 1) = f_5(a, b, n_1) + (x_5(a, b, n_1) > y_5(a, b, n_1)) \times f_7(a, b, n_1), \\
& f_6(a, b, n_1 + 1) = f_6(a, b, n_1) + (x_5(a, b, n_1) > y_5(a, b, n_1)) \times f_8(a, b, n_1), \\
& f_7(a, b, n_1 + 1) = f_7(a, b, n_1) + (\neg x_5(a, b, n_1) > y_5(a, b, n_1)) \times f_5(a, b, n_1), \\
& f_8(a, b, n_1 + 1) = f_8(a, b, n_1) + (\neg x_5(a, b, n) > y_5(a, b, n_1)) \times f_6(a, b, n_1).
\end{aligned}
$$

We see that the set of equations about $\{f_1, ..., f_4\}$ and the ones about $\{f_5, ..., f_8\}$ are isomorphic: $f_1 \leftrightarrow f_5$, $f_4 \leftrightarrow f_7$, $f_2 \leftrightarrow f_6$, $f_3 \leftrightarrow f_8$. In particular, we have

$$
\begin{aligned}
f_1(a, b, n_1) + f_4(a, b, n_1) &= f_5(a, b, n_1) + f_7(a, b, n_1), \\
f_2(a, b, n_1) + f_3(a, b, n_1) &= f_6(a, b, n_1) + f_8(a, b, n_1).
\end{aligned}
$$

When the program terminates, we have $x_5(a, b, N_1(a, b)) = y_5(a, b, N_1(a, b))$, thus

$$f_1(a,b,N_1(a,b)) \times x_5(a,b,0) - f_2(a,b,N_1(a,b)) \times y_5(a,b,0)$$
$$= f_3(a,b,N_1(a,b)) \times y_5(a,b,0) - f_4(a,b,N_1(a,b)) \times x_5(a,b,0).$$

This is the case for all possible a and b. So we have

$$(f_1(a,b,N_1(a,b)) + f_4(a,b,N_1(a,b))) \times a$$
$$= (f_2(a,b,N_1(a,b)) + f_3(a,b,N_1(a,b))) \times b,$$
$$(f_1(a-b,b,N_1(a-b,b)) + f_4(a-b,b,N_1(a-b,b))) \times (a-b)$$
$$= (f_2(a-b,b,N_1(a-b,b)) + f_3(a-b,b,N_1(a-b,b))) \times b.$$

Thus

$$u_5(a,b,N_1(a,b)) + v_5(a,b,N_1(a,b))$$
$$= (f_5(a,b,N_1(a,b)) + f_7(a,b,N_1(a,b)))a + (f_6(a,b,N_1(a,b)) + f_8(a,b,N_1(a,b)) \times b$$
$$= (f_1(a,b,N_1(a,b)) + f4(a,b,N_1(a,b)))a + (f_2(a,b,N_1(a,b)) + f_3(a,b,N_1(a,b))) \times b$$
$$= 2 \times (f_1(a,b,N_1(a,b)) + f_4(a,b,N_1(a,b))) \times a.$$

Similarly,

$$u_5(a-b,b,N_1(a-b,b)) + v_5(a-b,b,N_1(a-b,b))$$
$$= 2 \times (f_1(a-b,b,N_1(a-b,b))$$
$$+ f_4(a-b,b,N_1(a-b,b))) \times (a-b). \quad (6)$$

Now suppose $a > b$, by (2), and simultaneous induction on f_1 and f_4, it is easy to see that for all n_1, $f_1(a,b,n_1{+}1) = f_1(a{-}b,b,n_1)$ and $f_4(a,b,n_1{+}1) = f_4(a-b,b,n_1)$ (it is not the case that $f_2(a,b,n_1+1) = f_2(a-b,b,n_1)$). Thus by (3), if $a > b$, then

$$u_5(a,b,N_1(a,b)) + v_5(a,b,N_1(a,b))$$
$$= 2 \times (f_1(a,b,N_1(a,b)) + f_4(a,b,N_1(a,b))) \times a$$
$$= 2 \times (f_1(a,b,N_1(a,b)+1) + f_4(a,b,N_1(a,b)+1)) \times a$$
$$= 2 \times (f_1(a-b,b,N_1(a-b,b)) + f_4(a-b,b,N_1(a-b,b))) \times a.$$

From this and (6), (5) follows.

References

[1] F. Lin, "A formalization of programs in first-order logic with a discrete linear order," *Artificial Intelligence*, vol. 235, pp. 1–25, 2016.

[2] T. Jebelean, V. Negru, D. Petcu, D. Zaharie, T. Ida, and S. M. Watt, Eds., *19th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2017, Timisoara, Romania, September 21-24, 2017*. IEEE Computer Society, 2017. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8528958

[3] P. Rajkhowa and F. Lin, "Extending viap to handle array programs," in *10th Working Conference on Verified Software: Theories, Tools, and Experiments, VSTTE 2018, Oxford, UK, July 18-19*, 2018. [Online]. Available: https://github.com/VerifierIntegerAssignment/sv-comp/blob/master/extending-viap-array.pdf