Correctness And Run Time Complexity of Iterative Algorithms

1 Mathematical Induction

Definition 1: Mathematical Induction can prove a theorem T, which is true for integers starting from n_0 (usually 1, but may be any).

Steps:

1. Base case: T holds for n_0.
2. Inductive proof: Assume T is true for $n = k$, and prove that T also holds for $n = k + 1$.

Example $1 + 2 + ... + n = \left(\frac{n(n+1)}{2}\right)$
Prove by mathematical induction:

- **Base case:** True for $n = 1$, since $1 = 1\frac{(1+1)}{2}$.
- **Inductive proof:** Assume the claim is true for $n = 1, 2, \ldots, k$. Show that $n + 1$ holds because n holds:

$$
\begin{aligned}
1 + \ldots + k + (k+1) &\stackrel{?}{=} \frac{(k+1)(k+2)}{2} \\
\frac{k(k+1)}{2} + k + 1 &= \frac{k^2 + 3k + 2}{2} \\
\frac{k^2 + 3k + 2}{2} &= \frac{k^2 + 3k + 2}{2}
\end{aligned}
$$

- **Conclusion:** Holds for all $n \geq 1$

2 Correctness of Iterative Algorithms

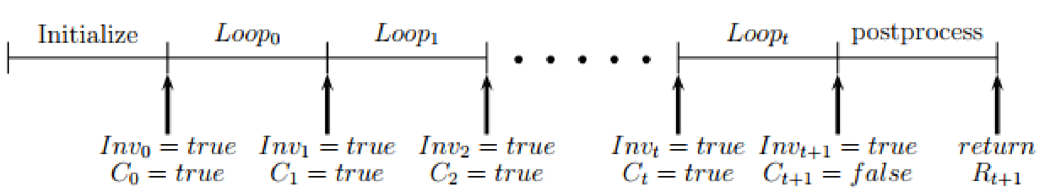
Definition 2: A **loop invariant** is a property which is related to the variables in a loop, and is true at the beginning of each iteration.

Proving Correctness of an Iterative Algorithms:

1. State the loop invariant.
2. Prove that invariant holds for any number of iterations using mathematical induction:
   (a) Prove that the loop invariant's base case holds – use values with index 0, that means use initialization values.
   (b) Prove that if invariant holds after k iterations it will hold after $k+1$.
3. Prove that the loop terminates.
4. Prove the correctness of the return value.

Here is a diagram of the algorithm execution:

- **Inv** is the invariant
- **C** is the loop condition
- enumerate iteration starting index 0 – in which case index i means "the value of the local variable (or invariant, or loop conditional) after i iterations of the loop".
- then during the i iteration (marked as $Loop_i$ on the diagram) the indices of the local variables change from i to $i+1$.
- We only look at values "in between" iterations – think during the while-loop condition checks and during the return statement. Those position are marked with tick on the diagram.
- the index t is the index of the last iteration, during which indices of the local variables change from t to $t+1$. Loop condition fails during the next check (C_{t+1} is false) and we go to code after the loop. Remember – all variable have indices $t+1$.



Example Prove that $ALG1(A,B)$ returns AB

```
ALG1(A, B) // A,B are natural numbers
{
    S = 0
    I = 0
    while (I < B)
    {
        S = S + A
        I = I + 1
    }
    return S
}
```

1. State the loop invariant:

   $I_k = k \;\; AND \;\; S_k = I_k A$ where index k is the number of **completed** iterations.

2. Prove that the loop invariant's base case holds. Base case refers to the values of the variable **before** the first iteration, that is all variable still have the initial values ($I_0 = 0$ and $S_0 = 0$). Corresponding index is 0 – no iteration have been completed:

$$\begin{aligned} I_0 &= 0 \\ S_0 &= I_0 A \end{aligned}$$

   since $I_0 = 0$ and $S_0 = 0$, the above equations hold.

3. Prove the invariant holds for some arbitrary iteration: assume invariant holds for all indices $1, \dots, k$ and prove that it will also hold for index $k+1$. That is – assume $S_k = kA$ and $I_k = k$ and prove $S_{k+1} = (k+1)A$ and $I_{k+1} = k+1$

   Proof:

$$\begin{array}{rcl|rcl} I_{k+1} & \stackrel{?}{=} & k+1 & S_{k+1} & \stackrel{?}{=} & (k+1)A \\ I_k + 1 & = & & S_k + A & = & \\ k+1 & = & & kA + A & = & \\ & & & (k+1)A & = & \end{array}$$

4. Prove that the loop terminates:

   Note: We will use the following mathematical theorem: a strictly increasing sequence of integers cannot be bounded from above.

   Consider I_k. As we have shown $I_k = k$, so it is a strictly increasing sequence of integers. Thus it cannot be bounded, or in other words $I_k < B$ cannot be true forever. So – the loop will eventually terminate, which means there is an index t so that loop condition holds for all indices $0, \ldots, t$, but not for $t+1$:

$$
\begin{array}{rcl}
I_0 & < & B \\
I_1 & < & B \\
\ldots & < & \ldots \\
I_t & < & B \\
I_{t+1} & \geq & B
\end{array}
$$

5. Prove the correctness of the return value. Notice that since the last iteration index is t, the indices of local variable after the loop terminates are $t+1$. So to show correctness of the return value we have to show $S_{t+1} = A \times B$:

   From the previous step, we know that a t exists such that:

$$
\begin{array}{rcl}
I_t & < & B \\
I_{t+1} & \geq & B
\end{array}
$$

   Solving the 2 inequalities and using the fact that $I_t = t$ (from loop invariant) and t is natural number, we get $t = B - 1$. Substitute $t = B - 1$ into the invariant:

$$
\begin{array}{rcl}
S_{t+1} & = & (t+1) \times A \\
 & = & ((B-1)+1) \times A \\
 & = & A \times B
\end{array}
$$

Example Prove the fast exponentiation function $FE(A, M)$ returns A^M.

```
FE(A,M)
{
    B = A;
    E = M;
    R = 1;
    while(E > 0)
    {
        if(E is odd)
        {
            R = R * B;
            E = E - 1;
        }
        else
        {
            B = B * B;
            E = E / 2;
        }
    }
    return R;
}
```

1. State the loop invariant:

   $A^M = R_k B_k^{E_k}$ where k is the some iteration index.

2. Prove that the loop invariant's base case holds:

$$\begin{aligned} k &= 0 \\ A^M &\stackrel{?}{=} R_0 B_0^{E_0} \\ &= (1)(A)^{(M)} \\ &= A^M \end{aligned}$$

3. Prove the invariant holds for some arbitrary iteration:

When E is odd:	When E is even:
$R_{k+1}B_{k+1}^{E_{k+1}} =$	$R_{k+1}B_{k+1}^{E_{k+1}} =$
$(R_kB_k)(B_k)^{(E_k-1)} =$	$(R_k)(B_kB_k)^{(\frac{E_k}{2})} =$
$R_kB_k^{E_k} =$	$R_kB_k^{2(\frac{E_k}{2})} =$
$A^M =$	$R_kB_k^{E_k} =$
	$A^M =$

4. Prove that the loop terminates:
   Note: A strictly decreasing sequence of integers cannot be bounded.
   There are two ways E_k decreases:

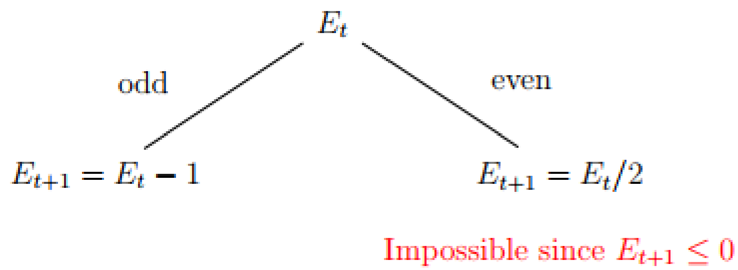
   (a) $E = E - 1$
   (b) $E = E/2$ (since $E > 0$ given the loop condition)

   Since E decreases in both cases and the loop condition is $E > 0$, the loop will terminate.

5. Prove the correctness of the return value (Show $A^M = R_kB_k^{E_k}$):

$$\begin{aligned} E_0 &> 0 \\ E_1 &> 0 \\ &\vdots \\ E_t &> 0 \\ E_{t+1} &\leq 0 (Loop\ terminates) \\ \therefore A^M &= R_{t+1}B_{t+1}^{E_{t+1}} \end{aligned}$$

   To prove this, we look at the last iteration, E_t:



$$\begin{cases} E_t & > 0 \\ E_t - 1 & \leq 0 \end{cases}$$

$$\begin{cases} E_t & > 0 \\ E_t & \leq 1 \end{cases}$$

$\therefore E_t = 1 \rightarrow E_{t+1} = 0 \rightarrow A^M = R_k B_k^0 = R_k$