CSC236 winter 2020, week 8: Proving termination

Recommended supplementary reading: Chapter 2 Vassos course notes

Colin Morris
colin@cs.toronto.edu
http://www.cs.toronto.edu/~colin/236/W20/

March 2, 2020

Proving termination

```python
def imax(A):
  """Pre: A is non-empty and contains comparable items.
  Post: return the maximum element in A
  """
  curr = A[0]
  i = 1
  while i < len(A):
    if A[i] > curr:
      curr = A[i]
    i += 1
  return curr
```

Eventually i must reach `len(A)`...

A corollary of principle of well-ordering

All decreasing sequences of natural numbers are finite.

Spot the decreasing sequence

```
def imax(A):
  """Pre: A is non-empty and contains comparable items.
  Post: return the maximum element in A
  """
  curr = A[0]
  i = 1
  while i < len(A):
    if A[i] > curr:
      curr = A[i]
    i += 1
  return curr
```

$len(A) - i_j$

decreasing?

in $\mathbb{N}$? yes, but need an invariant...

$i_j \leq len(A)$

Another corollary of PWO

Any increasing sequence of natural numbers with an upper bound is finite.

Recipe: proving termination

Define some quantity m_j associated with each iteration j of the loop.

- will be defined in terms of one or more variables that change inside the loop
- e.g. $m_j = \text{len}(A) - i_j$

Show that

- Every $m_j \in \mathbb{N}$
- the sequence $\langle m_0, m_1, m_2, \ldots \rangle$ is decreasing.

Example: A2Q3 appendix

```
1   def R(A):
2       B = []
3       i = 0
4       while i < len(A):
5           a = A[i]
6           b = A[(i+1) % len(A)]
7           if a == b:
8               B.append(a)
9           i += 1
10      return B
```

Inv: $i_j \le \text{len}(A)$

Lemma (R termination)

R terminates on any $A \in \mathbb{N}^*$

Proof.

Let $m_j = \text{len}(A) - i_j$ be a quantity associated with each loop iteration j. By Lemma 1.3 (a), $m_j \in \mathbb{N}$. By line 9, $m_{j+1} = m_j - 1$. Thus $m_0, m_1, m_2, \ldots$ is a decreasing sequence of natural numbers, and therefore finite. Therefore, R terminates. □

(optional conclusion)

merge

```python
def merge(A, B):
  """Pre: A and B are sorted lists of numbers.
  Post: return a sorted permutation of A+B
  """
  i = j = 0
  C = []
  while i < len(A) and j < len(B):
    if A[i] <= B[j]:
      C.append(A[i])
      i += 1
    else:
      C.append(B[j])
      j += 1
  return C + A[i:] + B[j:]
```

× $m_k = \min(\mathrm{len}(A) - i_k,\ \mathrm{len}(B) - j_k)$

$m_k = (\mathrm{len}(A) + \mathrm{len}(B)) - (i_k + j_k)$

decreasing? ✓

$\in \mathbb{N}$ ✓

w/ loop invs...

- $\mathrm{len}(A) - i_k \geq 0$
- $\mathrm{len}(B) - j_k \geq 0$

bitcount (week 7 tutorial exercise)

```python
def bitcount(n):
    """Pre: n is a positive int.
    Post: return the number of digits in the binary representation of n
    """
    i = 1
    while n > 1:
        n = n//2
        i += 1
    return i
```

$m_j = n_j$

$n_{j+1} < n_j$? yes because

$n_{j+1} = n_j // 2$

$n_j > n_j // 2$, because $n_j > 0$ by the while cond.

$n_j \in \mathbb{N}$? yes by precondition, + $n//2 \in \mathbb{N}$ for all $n \in \mathbb{N}$

or use loop inv: $n_j \in \mathbb{N}$

Tricky example

```
1   def mystery(n):
2     """Pre: n is a positive int
3     """
4     i = 0
5     while n > 1:
6       if n % 10 == 0:
7         n = n // 10
8       else:
9         n += 1
10      i += 1
11    return i
```

See Piazza for alternative proof (using decreasing sequence)

$P(q)$: loop terminates if $n_k = q$ for some k.

Let $q \in \mathbb{N}$

Assume $P(k)$ for all $k < q$

Assume exists some iter k, s.t. $n_k = q$

Case: $q = 10x$, $x \in \mathbb{N}$, $x \geq 1$

$n_{k+1} = x$, by l 6-7

by $P(x)$ we terminate!

Case 2: q not a multiple of 10, $q \% 10 = r$

$m_j = (n_j - 1) \% 10 + 10 + n_j$

$u_j = n_j \% 10$

$m_j = n_j + (20 - 2 \times u_j)$