CPSC 320: Intermediate Algorithm Design 2016 Summer Term 1

Tutorial 9

Prof. Nick Harvey *University of British Columbia*

1. **(Amortized Analysis)** It is possible to implement a **queue** using a pair of **stacks** as follows:
   - ENQUEUE(x) pushes x on $stack1$.
   - DEQUEUE() first checks to see if $stack2$ contains any elements. If so, it returns $stack2$.POP(). Otherwise, if first transfers every element from $stack1$ onto $stack2$ by calling $stack2$.PUSH($stack1$.POP()) as many times as necessary, and then it returns $stack2$.POP().

   Using amortized analysis, prove that the worst-case running time of any sequence of n ENQUEUE and DEQUEUE operations is in $O(n)$.

2. **(Amortized Analysis of a Code Segment)**

   Consider the following algorithm:

```
Algorithm Mysterious(array)
  accumulator ← 0
  for i ← 0 to length[array] - 1 do

    while (accumulator > 0 and compute(accumulator, array[i]) > 0)
      accumulator ← accumulator - 1

    if (array[i] is even) then
      accumulator ← accumulator + floor(log(i+1))

  return accumulator
```

   Use the potential method to prove that this algorithm runs in $O(n \log n)$ time where $n =$ `length[array]`. You may assume that the function `compute()` runs in $\Theta(1)$ time. Hints:
   - Think of the state of the algorithm at the end of the i^{th} iteration as D_i.
   - Use the value of `accumulator` at the end of the i^{th} iteration as the potential of D_i.

   Do not forget to show that Φ is a valid potential function.