CPSC 331 S22

Jaza Khan UCID 30119100

Assignment 2 - Proofs

Given that n is the number of unique nodes in a linked list, prove that the runtime of Floyd's Tortoise and Hare algorithm is in $O(n)$.

PROOF

We begin by analyzing the runtime of this algorithm using the **uniform cost criterion** in order to form a function $f(n)$.

Suppose this algorithm is executed with an input size n.

Lines 86-87 each require 1 step.

In order to analzye the number of steps used when executing the `while` loop on lines 90-103, we can use a *bound function* for the loop. This will allow us to find the upper bound for the number of times that the `while` loop body is executed before this execution of the loop ends.

The body of the `while` loop contains 8 executable steps, each of which cost 1 step except for the `do-while` loop on lines 97-100. This `do-while` loop contains 2 executable steps, each of cost 1. These will be executed *at most* the length of the cycle (`period`) times.

After the loop, only line 104 still has to be executed, which costs 1 step.

Thus, the number of steps executed for this algorithm *at most* is: $12n + 10$

We will now prove that the runtime of Floyd's Tortoise and Hare algorithm is in $O(n)$, **by application of the definition.**

By definition of $O(n)$, we are required to show that there exists a constant $c > 0$ and a constant $N_0 > 0$ such that $f(n) \leq cn$ for all $n \in \mathbb{R}$ and $n \geq N_0$.

Let $c = 25$ and $N_0 = 3$.

Let n be an *arbitrarily chosen element* in the range of f such that $n \geq N_0$. We must prove that the claim holds for this choice of n.

Then,

$12n + 10 \leq 12n + 10n = 25n$

since $3 \leq 3 \leq n$ whenever $n \geq 3$. Since n was arbitrarily chosen from $\mathbb{R}$, it follows that $12n + 10 \leq 12n + 10n \leq 25n = cn$ for all $n \in \mathbb{R}$ such that $n \geq 3 = N_0$. Since $c = 25$ and $N_0 = 3$ are constants, this establishes the claim that they are *existentially quantified*, as needed to conclude $12n + 10 \in O(n)$.

We have now proved a result for all n in the range of f such that $n \geq N_0$.

Another way of proving that Floyd's Hare and Tortoise algorithm is in linear time is using a limit test for $O(g)$.

By definition of the theorem, if $\lim_{x \to +\infty} \frac{f(x)}{g(x)}$ exists and is a real constant — so that in particular, it is not
equal to $+\infty$ — then $f \in O(g)$.

Let $f(n) = 12n + 10$ and $g(n) = n$. Then,

$\lim_{n \to +\infty} (\frac{12n+10}{n})$

$\lim_{n \to +\infty} (12 + \frac{10}{n})$

$= 12$

Since 12 is a real and non-negative constant, it now follows by the Limit Test for $O(g)$ that this algorithm is in $O(n)$, where n is the unique number of nodes in a linked list.

Given that n is the sum of the nodes in `list0` and `list`, prove that the runtime of the `removeSharedLinkedListNodes` algorithm is in $O(n)$.

PROOF

We begin by analyzing the runtime of this algorithm using the **uniform cost criterion** in order to form a function $f(n)$.

Suppose this algorithm is executed with an input size n.

Lines 29-30 each cost 1 step.

If `list0` is longer than `list1`, then lines 33-37 cost 1 step. If, however, `list1` is longer than `list0`, lines 33-37 will cost 4 steps due to the swapping process.

Line 40 costs 1 step.

Lines 43-44 cost 1 step each, as we are simply calling the `getHeadNode()` function.

The `for` loop on lines 47-49 contains one executable step in the body of the loop. This will be executed *at most* `lengthDiff` times, while the loop test itself will be executed *at most* `lengthDiff+1` times.

Line 51 costs 1 step.

In order to analzye the number of steps used when executing the `while` loop on lines 53-71, we can use a *bound function* for the loop. This will allow us to find the upper bound for the number of times that the `while` loop body is executed before this execution of the loop ends.

The body of this `while` loop contains 9 executable steps, each of which cost 1 step except for the `for` loop on lines 59-61. Similarly to the previous `for` loop, we can examine the worst case for this as well and the number of steps it would require *at most*.

The body of the `while` loop would execute `listLength0 + listLength1` number of times *at most*, while the loop test would execute that + 2 more steps.

Thus, the number of steps executed for this algorithm *at most* is: $14n + 9$

We will now prove that the runtime of the `removeSharedLinkedListNodes` algorithm is in $O(n)$, **by application of the definition.**

By definition of $O(n)$, we are required to show that there exists a constant $c > 0$ and a constant $N_0 > 0$ such that $f(n) \leq cn$ for all $n \in \mathbb{R}$ and $n \geq N_0$.

Let $c = 30$ and $N_0 = 1$.

Let n be an *arbitrarily chosen element* in the range of f such that $n \geq N_0$. We must prove that the claim holds for this choice of n. Then,

$14n + 9 \leq 14n + 9n = 30n$

since $1 \leq 1 \leq n$ whenever $n \geq 1$. Since n was arbitrarily chosen from $\mathbb{R}$, it follows that $14n + 9 \leq 14n + 9n \leq 30n = cn$ for all $n \in \mathbb{R}$ such that $n \geq 1 = N_0$. Since $c = 30$ and $N_0 = 1$ are constants, this establishes the claim that they are *existentially quantified*, as needed to conclude $14n + 9 \in O(n)$.

We have now proved a result for all n in the range of f such that $n \geq N_0$.