CS/ECE 374 P14

Jiawei Tang, Junquan Chen, Pengxu Zheng

TOTAL POINTS

50 / 100

QUESTION 1

1 Problem 14.A. 15 / 20

✓ - 5 pts **Missing explanation or largely incorrect justification of algorithm correctness**

QUESTION 2

2 Problem 14.B. 25 / 40

✓ - 10 pts **Missing explanation or largely incorrect justification of algorithm correctness**

✓ - 5 pts **Small error in algortihm, justification of algorithm, or number of calls calculations**

"Placeholders" should likely be lists that are appended to rather than strings that are assigned to.

QUESTION 3

3 Problem 14.C. 10 / 40

✓ - 30 pts **IDK**

HW5 Solution

CS/ECE 374: Algorithms & Models of Computation, Spring 2019 Version: **1.0**

Submitted by:

- **≪Pengxu Zheng≫**: **≪pzheng5≫**
- **≪Junquan Chen≫**: **≪junquan2≫**
- **≪Jiawei Tang≫**: **≪jiaweit2≫**

14

Solution:

14.A.

```
FindMissing (A[0, ..., n - 1]):
result = ""        // initialize result to an empty string
for j in l:
    num0 < − 0  // set the counter for the number of 0's to 0
    num1 < − 0  // set the counter for the number of 1's to 0
    for i in n:
        if FetchBit(i, j) == 0:
            num0++
        else:
            num1++
    if num0 < num1:
        result += '0'
    else:
        result += '1'
return result
```

Since $n = 2^l - 1$, $l = log2(n + 1)$. The total number of FetchBit calls is $n * l$, which is $nlog(n + 1)$. Thus, we conclude that our algorithm runs in O(nlogn) time.

14.B.

```
FindMissing (A[0, ..., n - 1]):
result = ""        // initialize result to an empty string
partial0, partial1 = ""    // declare two partial placeholders for A.
for j = l - 1; j >= 0; j–:
    partial0, partial1 = ""  // reset the partial placeholders
    for i = 0; i < strlen(A); i++:
        if FetchBit(i, j) == 0:
            partial0 = A[i]
        else:
            partial1 = A[i]
    if strlen(partial1) < strlen(partial0):
        result += '0'
        A = partial1
    else:
        result += '1'
        A = partial0
```

1 Problem 14.A. 15 / 20

✓ **- 5 pts** **Missing explanation or largely incorrect justification of algorithm correctness**

HW5 Solution

CS/ECE 374: Algorithms & Models of Computation, Spring 2019 Version: **1.0**

Submitted by:

- **≪Pengxu Zheng≫**: **≪pzheng5≫**
- **≪Junquan Chen≫**: **≪junquan2≫**
- **≪Jiawei Tang≫**: **≪jiaweit2≫**

14

Solution:

14.A.

```
FindMissing (A[0, ..., n - 1]):
result = ""        // initialize result to an empty string
for j in l:
    num0 < − 0  // set the counter for the number of 0's to 0
    num1 < − 0  // set the counter for the number of 1's to 0
    for i in n:
        if FetchBit(i, j) == 0:
            num0++
        else:
            num1++
    if num0 < num1:
        result += '0'
    else:
        result += '1'
return result
```

Since $n = 2^l - 1$, $l = log2(n+1)$. The total number of FetchBit calls is $n * l$, which is $nlog(n+1)$. Thus, we conclude that our algorithm runs in O(nlogn) time.

14.B.

```
FindMissing (A[0, ..., n - 1]):
result = ""        // initialize result to an empty string
partial0, partial1 = ""   // declare two partial placeholders for A.
for j = l - 1; j >= 0; j–:
    partial0, partial1 = ""  // reset the partial placeholders
    for i = 0; i < strlen(A); i++:
        if FetchBit(i, j) == 0:
            partial0 = A[i]
        else:
            partial1 = A[i]
    if strlen(partial1) < strlen(partial0):
        result += '0'
        A = partial1
    else:
        result += '1'
        A = partial0
```

2 Problem 14.B. **25 / 40**

✓ **- 10 pts Missing explanation or largely incorrect justification of algorithm correctness**

✓ **- 5 pts Small error in algortihm, justification of algorithm, or number of calls calculations**

"Placeholders" should likely be lists that are appended to rather than strings that are assigned to.

```
    return result
```

Since each time we cut the string in half, the total number of FetchBit calls is $2^l - 1 + 2^{l-1} - 1 + ... + 2^1 - 1$, which is less than 2^{l+1} calls and that is also less than 2(n + 1) calls. Hence, we conclude that is an O(n) algorithm.

14.C.
IDK. sorry

3 Problem 14.C. **10 / 40**

✓ **- 30 pts** IDK