Problem Set III

Huy Quang Lai
132000359

Texas A&M University

26 November 2022

Huffman Encoding

Problem 1

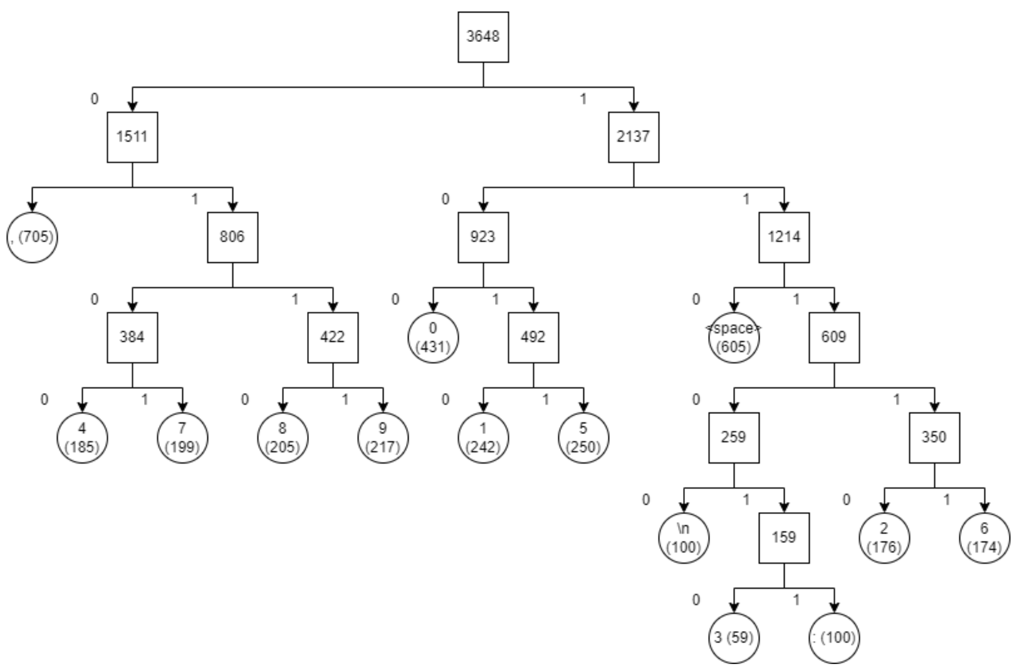

Figure 1: Huffman Tree

Character	Code
,	00
<space>	110
0	100
5	1011
1	1010
9	0111
8	0110
7	0101
4	0100
2	11111
6	11110
\n	11100
:	111011
3	111010

Problem 2

If the symbols are sorted by most frequent to least frequent, then Huffman encoding is a $O(n)$ algorithm.
Starting from the second symbol, create a new internal node based on the sum of this symbol and the previous symbol's frequency. Continue created new internal nodes until all symbols are visited.
This algorithm will traverse $n-1$ symbols, therefore this algorithm is $O(n)$

Karatsuba Multiplication

Problem 1

```
string karatsuba(string lhs, string rhs) {
    size_t length = max(lhs.size(), rhs.size());

    while (lhs.size() < length)
        lhs.insert(0, "0");

    while (rhs.size() < length)
        rhs.insert(0, "0");

    if (length == 1)
        return to_string((lhs[0] - '0') * (rhs[0] - '0'));

    // Split lhs and rhs into smaller strings
    string lhs0 = lhs.substr(0, length / 2);
    string lhs1 = lhs.substr(length / 2, length - length / 2);
    string rhs0 = rhs.substr(0, length / 2);
    string rhs1 = rhs.substr(length / 2, length - length / 2);

    string p0 = multiply(lhs0, rhs0);   // z0
    string p1 = multiply(lhs1, rhs1);   // z2
    // z1 = (x1+x0)(y1+y0)
    string p2 = multiply(add(lhs0, lhs1), add(rhs0, rhs1));

    // z1 = (x1+x0)(y1+y0)-(z2+z0)
    string p3 = subtract(p2, add(p0, p1));

    // multiply by 10^(2m2)
    for (size_t i = 0; i < 2 * (length - length / 2); i++)
        p0.append("0");
    // multiply by 10^(m2)
    for (size_t i = 0; i < length - length / 2; i++)
        p3.append("0");
```

```
    // final steps of the algorithm
    string result = add(add(p0, p1), p3);
    return result.erase(0,
    min(result.find_first_not_of('0'), result.size() - 1));
}
```

$117937 = 117 \times (10^3) + 937$
$404783 = 404 \times (10^3) + 783$
$z_2 = 117 \times 404 = 47268$
$z_0 = 937 \times 783 = 733671$
$z_1 = (117 + 937) \times (404 + 783) - 47268 - 733671 = 470159$
Result $= 47268 \times (10^3)^2 + 470159 \times (10^3) + 733671 = 47738892671$

Recursive steps:
For z_2 :
$117 = 1 \times (10^2) + 17$
$404 = 4 \times (10^2) + 4$
$z_2 = 1 \times 4 = 4$
$z_0 = 17 \times 4 = 68$
$z_1 = (1 + 17) \times (4 + 4) - 4 - 68 = 72$
Result $= 4 \times (10^2)^2 + 72 \times (10^2) + 68 = 47268$

For $z_1 \rightarrow (117 + 937) \times (404 + 783)$:
$1054 = 10 \times (10^2) + 54$
$1187 = 11 \times (10^2) + 87$
$z_2 = 10 \times 11 = 110$
$z_0 = 54 \times 87 = 4698$
$z_1 = (10 + 54) \times (11 + 87) - 110 - 4698 = 1464$
Result $= 110 \times (10^2)^2 + 1464 \times (10^2) + 4698 = 1251098$

For z_0 :
$937 = 9 \times (10^2) + 37$
$783 = 7 \times (10^2) + 83$
$z_2 = 9 \times 7 = 63$
$z_0 = 37 \times 83 = 3071$
$z_1 = (9 + 37) \times (7 + 83) - 63 - 3071 = 1006$
Result $= 63 \times (10^2)^2 + 1006 \times (10^2) + 3071 = 733671$

Problem 2

The algorithm recursively calls itself with half the string, therefore $T\left(\frac{n}{2}\right)$ is needed.
Since the algorithm recursively calls itself three times, $3T\left(\frac{n}{2}\right)$.
Additionally, both subtracting and addition are $O(n)$ processes.
Therefore, the algorithm has a total of $3T\left(\frac{n}{2}\right) + O(n)$.
Using the master theorem, this can be simplified to $O\left(n^{\log_2 3}\right)$.