



WYDZIAŁ  
INFORMATYKI  
I TELEKOMUNIKACJI

# Obiegi drzew

Veronika Tronchuk

I rok Informatyki

Nr albumu: 30019

# Celem

ćwiczenia było zapoznanie się z działaniem algorytmu Huffmana, który umożliwia efektywną kompresję danych tekstowych poprzez przypisanie krótszych kodów częściej występującym znakom. W ramach zadania zaimplementowano budowę drzewa Huffmana, generowanie kodów, kodowanie i dekodowanie tekstu oraz obliczenie współczynnika kompresji.

```
#include <iostream>
#include <queue>
#include <unordered_map>
#include <vector>
```

Dołączam biblioteki: iostream do wejścia/wyjścia, queue do kolejki węzłów drzewa, unordered\_map do mapy częstotliwości i kodów, vector do tablic.

```
struct Node {
    char character;
    int frequency;
    Node* left;
    Node* right;
```

Struktura Node — węzeł drzewa: przechowuje znak (character), częstotliwość (frequency) i wskaźniki na lewego i prawego potomka.

```
Node(char ch, int freq)
: character(ch), frequency(freq), left(nullptr), right(nullptr) {}

Node(char ch, int freq, Node* left, Node* right)
: character(ch), frequency(freq), left(left), right(right) {}
```

Pierwszy konstruktor — jeśli węzeł to liść (bez dzieci). Drugi — jeśli ma dzieci.

```
struct compare {
    bool operator()(Node* left, Node* right) {
        return left->frequency > right->frequency;
    }
};
```

Ta struktura compare określa, jak sortować węzły w kolejce: węzeł z mniejszą częstotliwością ma wyższy priorytet.

```
void printCodes(Node* root, string str, unordered_map<char, string>& huffmanCode) {  
    if (root == nullptr)  
        return;  
  
    if (!root->left && !root->right) {  
        huffmanCode[root->character] = str;  
    }  
  
    printCodes(root->left, str + "0", huffmanCode);  
    printCodes(root->right, str + "1", huffmanCode);  
}
```

Funkcja printCodes: jeśli węzła nie ma — wychodzę. Jeśli to liść — zapisuję kod dla znaku. Dalej rekurencyjnie idę w lewo (+ "0") i w prawo (+ "1").

```
void buildHuffmanTree(string text) {  
    unordered_map<char, int> freq;  
    for (char ch : text) {  
        freq[ch]++;  
    }  
}
```

Zliczam częstotliwość znaków i zapisuję w freq

```
priority_queue<Node*, vector<Node*>, compare> pq;  
for (auto pair : freq) {  
    pq.push(new Node(pair.first, pair.second));  
}
```

Tworzę kolejkę i dodaję wszystkie znaki z częstotliwościami.

```
while (pq.size() != 1) {  
    Node* left = pq.top(); pq.pop();  
    Node* right = pq.top(); pq.pop();  
    int sum = left->frequency + right->frequency;  
    pq.push(new Node('\0', sum, left, right));  
}
```

Dopóki w kolejce więcej niż jeden węzeł: biorę dwa z najmniejszą częstotliwością, tworzę nowego „rodzica” i odkładam.

```
Node* root = pq.top();
```

Teraz w kolejce został tylko korzeń drzewa.

```
unordered_map<char, string> huffmanCode;  
printCodes(root, "", huffmanCode);
```

Tworzę tabelę kodów i wypełniam przez printCodes.

```
cout << "Huffman Codes:\n";  
for (auto pair : huffmanCode) {  
    cout << "Character: " << pair.first << ", frequency: " << freq[pair.first]  
        << ", Huffman code: " << pair.second << "\n";  
}
```

Dla każdego znaku pokazuję: znak, ile razy jest w tekście i jaki ma kod Huffmana.

```
cout << "\nOriginal string:\n" << text << "\n";  
  
string str = "";  
for (char ch : text) {  
    str += huffmanCode[ch];  
}  
cout << "\nEncoded string:\n" << str << "\n";
```

Pokazuję oryginalny tekst i zakodowany jako ciąg zer i jedynek.

```
auto decode = [&](string str) {  
    cout << "\nDecoded string:\n";  
    Node* curr = root;  
    for (char bit : str) {  
        if (bit == '0') curr = curr->left;  
        else curr = curr->right;  
        if (!curr->left && !curr->right) {  
            cout << curr->character;  
            curr = root;  
        }  
    }  
    cout << "\n";  
};
```

Dekoder: czyta bity i odtwarza oryginalny tekst z drzewa.

```
int bityASCII = text.length() * 8;
int bityHuffmana = 0;
for (auto pair : freq) {
    bityHuffmana += pair.second * huffmanCode[pair.first].length();
}
double wspKompresji = (double)bityASCII / bityHuffmana;

cout << "\nBits ASCII: " << bityASCII << endl;
cout << "Bits Huffmana: " << bityHuffmana << endl;
cout << "Compression ratio: " << wspKompresji << endl;
```

Obliczam ile bitów zajmuje normalnie (ASCII) i ile z Huffmanem. Potem pokazuję współczynnik kompresji.

```
int main() {
    string text = R"(Zasadził dziadek rzepkę w ogrodzie,
        Chodził tę rzepkę oglądać co dzień
        Kawka na trawkę. )"
    ;
    buildHuffmanTree(text);
```

Podaję tekst (Rzepka) i uruchamiam drzewo Huffmana

## „Rzepka”

Za-sa-dził dia-dek rzep-kę w ogro-dzie,  
Cho-dził tę rzep-kę oglą-dać co dzień.  
Wy-ro-sła rzep-ka jędr-na i krzep-ka,  
Schru-pać by rzep-kę z ka-wał-kiem chleb-ka!  
Więc cią-gnie rzep-kę dia-dek nie-bo-żę,  
Cią-gnie i cią-gnie, wy-cią-gnąć nie może.

Za-wo-łał dia-dek na pomoc bab-cię:  
„Ja zła-pię rzecz-kę, ty za mnie złap się!”  
I bied-ny dia-dek z bab-cią nie-bo-gą

Cią-gną i cią-gną, wy-cią-gnąć nie mogą.  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
Oj, przy-dał-by się ktoś na przy-czep-kę!

Przy-le-ciał wnu-czek, babci się zła-pał,  
Poci się, stęka, aż się za-sa-pał!  
Wnu-czek za bab-cię,  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
Oj, przy-dał-by się ktoś na przy-czep-kę!  
Pocą się, sapią, stę-ka-ją srogo,  
Cią-gną i cią-gną, wy-cią-gnąć nie mogą.

Za-wo-łał wnu-czek szcze-niacz-ka Mrucz-ka,  
Przy-le-ciał Mru-czek i cią-gnie wnucz-ka!  
Mru-czek za wnucz-ka,  
Wnu-czek za bab-cię,  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
Oj, przy-dał-by się ktoś na przy-czep-kę!  
Pocą się, sapią, stę-ka-ją srogo,  
Cią-gną i cią-gną, wy-cią-gnąć nie mogą!

Na kurkę czy-hał kotek w ukry-ciu,  
Za-szcze-kał Mru-czek: „Pomóż nam, Kiciu!”  
Kicia za Mrucz-ka,  
Mru-czek za wnucz-ka,  
Wnu-czek za bab-cię,  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
Oj, przy-dał-by się ktoś na przy-czep-kę!  
Pocą się, sapią, stę-ka-ją srogo,  
Cią-gną i cią-gną, wy-cią-gnąć nie mogą!

Więc woła Kicia kurkę z po-dwór-ka,  
Wnet przy-le-cia-ła usług-na kurka.  
Kurka za Kicię,  
Kicia za Mrucz-ka,  
Mru-czek za wnucz-ka,  
Wnu-czek za bab-cię,  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
Oj, przy-dał-by się ktoś na przy-czep-kę!  
Pocą się, sapią, stę-ka-ją srogo,  
Cią-gną i cią-gną, wy-cią-gnąć nie mogą!

Szła sobie gąska ście-ży-ną wąską,  
Krzyk-nę-ła kurka: „Chodź no tu, gąsko!”  
Gąska za kurkę,  
Kurka za Kicię,  
Kicia za Mrucz-ka,  
Mru-czek za wnucz-ka,  
Wnu-czek za bab-cię,  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
Oj, przy-dał-by się ktoś na przy-czep-kę!  
Pocą się, sapią, stę-ka-ją srogo,  
Cią-gną i cią-gną, wy-cią-gnąć nie mogą! Le-ciał wy-so-ko bocian-  
długonos,  
„Fruń-że tu, boćku, do nas na pomoc!”  
Bo-ciek za gąskę,  
Gąska za kurkę,  
Kurka za Kicię,  
Kicia za Mrucz-ka,  
Mru-czek za wnucz-ka,  
Wnu-czek za bab-cię,  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
Oj, przy-dał-by się ktoś na przy-czep-kę!

Pocą się, sapią, stę-ka-ją srogo,  
Cią-gną i cią-gną, wy-cią-gnąć nie mogą!

Ska-ka-ła drogą zie-lo-na żabka,  
Zła-pa-ła boćka – rzad-ka to grat-ka!  
Żabka za boćka,  
Bo-ciek za gąskę,  
Gaska za kurkę,  
Kurka za Kicię,  
Kicia za Mrucz-ka,  
Mru-czek za wnucz-ka,  
Wnu-czek za bab-cię,  
Bab-cia za dziad-ka,  
Dzia-dek za rzep-kę,  
A na przy-czep-kę  
Kawka za żabkę,  
Bo na tę rzep-kę  
Też miała chrap-kę.

Tak się za-wzię-li,  
Tak się na-dę-li,  
Że nagle rzep-kę  
Trr-rach!! – wy-cią-gnę-li!  
Aż wstyd po-wie-dzieć,  
Co było dalej!  
Wszy-scy na sie-bie  
Po-upa-da-li:

Rzep-ka na dziad-ka,  
Dzia-dek na bab-cię,  
Bab-cia na wnucz-ka,  
Wnu-czek na Mrucz-ka,  
Mru-czek na Kicię,  
Kicia na kurkę,  
Kurka na gąskę,  
Gaska na boćka,



Bo-ciek na żabkę,  
Żabka na kawkę  
I na ostat-ku  
Kawka na traw-kę.

<https://zwierciadlo.pl/kultura/548398,1,rzepka--wiersz-juliana-tuwima-ktory-bawi-dzieci-od-pokolen.read>

```
Huffman Codes:
Character: 'o', frequency: 78, Huffman code: 00000
Character: 'Z', frequency: 5, Huffman code: 001100000
Character: ':', frequency: 4, Huffman code: 1000010011
Character: 'z', frequency: 172, Huffman code: 0001
Character: 'r', frequency: 86, Huffman code: 00001
Character: 'I', frequency: 2, Huffman code: 0011000010
Character: 'i', frequency: 178, Huffman code: 0010
Character: 'C', frequency: 11, Huffman code: 00110001
Character: 'S', frequency: 3, Huffman code: 0011000011
Character: 'P', frequency: 11, Huffman code: 00110010
Character: ' ', frequency: 108, Huffman code: 01010
Character: 'l', frequency: 12, Huffman code: 00110011
Character: 'y', frequency: 47, Huffman code: 001101
Character: '.', frequency: 7, Huffman code: 010110100
Character: 'n', frequency: 95, Huffman code: 00111
Character: 'a', frequency: 269, Huffman code: 1010
Character: '!', frequency: 24, Huffman code: 0100000
Character: 'B', frequency: 13, Huffman code: 01000010
Character: 'W', frequency: 13, Huffman code: 01000011
Character: 'g', frequency: 52, Huffman code: 010001
Character: 'e', frequency: 102, Huffman code: 01001
Character: 't', frequency: 27, Huffman code: 0101100
Character: 'h', frequency: 7, Huffman code: 010110101
Character: '-', frequency: 1, Huffman code: 100001110100
Character: 'i', frequency: 247, Huffman code: 0111
Character: 'm', frequency: 15, Huffman code: 01011011
Character: 'b', frequency: 56, Huffman code: 010111
Character: 'J', frequency: 1, Huffman code: 100001110101
Character: ' ', frequency: 118, Huffman code: 01100
Character: 'p', frequency: 59, Huffman code: 011010
Character: 'd', frequency: 60, Huffman code: 011011
Character: 'u', frequency: 60, Huffman code: 100000
Character: 'O', frequency: 7, Huffman code: 100001000
Character: 'G', frequency: 4, Huffman code: 1000010010
Character: 'j', frequency: 15, Huffman code: 10000101
Character: 'M', frequency: 16, Huffman code: 10000110
Character: 'û', frequency: 2, Huffman code: 10000111000
Character: 'A', frequency: 2, Huffman code: 10000111001
Character: 'L', frequency: 1, Huffman code: 100001110110
Character: 'R', frequency: 1, Huffman code: 100001110111
Character: 'ö', frequency: 4, Huffman code: 1000011111
Character: 'T', frequency: 4, Huffman code: 1000011110
Character: 'c', frequency: 126, Huffman code: 10001
Character: '?', frequency: 255, Huffman code: 1001
Character: 's', frequency: 2, Huffman code: 10110110001
Character: 's', frequency: 63, Huffman code: 101100
Character: 'w', frequency: 35, Huffman code: 1011010
Character: 'F', frequency: 1, Huffman code: 101101100000
Character: 'N', frequency: 1, Huffman code: 101101100001
Character: 'D', frequency: 9, Huffman code: 101101101
Character: 'ä', frequency: 4, Huffman code: 1011011001
Character: 'K', frequency: 21, Huffman code: 10110111
Character: 'k', frequency: 164, Huffman code: 10111
Character: ' ', frequency: 1201, Huffman code: 11
```

Original string:

Za;sa;dzi? dzia;dek rzep;k? w ogro;dzie,  
Cho;dzi? t? rzep;k? ogl?;da? co dzie?..  
Wy;ro;s?a rzep;ka j?dr;na i krzep;ka,  
Schru;pa? by rzep;k? z ka;wa?;kiem chleb;ka!  
Wi?c ci?;gnie rzep;k? dzia;dek nie;bo;??,  
Ci?;gnie i ci?;gnie, wy;ci?;gn?? nie mo?e.

Za;wo;?a? dzia;dek na pomoc bab;ci? :  
äJa z?a;pi? rzec;ki?, ty za mnie z?ap si?!ö  
I bied;ny dzia;dek z bab;ci? nie;bo;g?  
Ci?;gn? i ci?;gn?, wy;ci?;gn?? nie mog?..  
Bab;cia za dziad;ka,  
Dzia;dek za rzep;k?,  
Oj, przy;da?;by si? kto? na przy;czep;k?!

Przy;le;cia? wnu;czek, babci si? z?a;pa?,  
Poci si?, st?ka, a? si? za;sa;pa?!  
Wnu;czek za bab;ci?,  
Bab;cia za dziad;ka,  
Dzia;dek za rzep;k?,  
Oj, przy;da?;by si? kto? na przy;czep;k?!  
Poc? si?, sapi?, st?;ka;j? srogo,  
Ci?;gn? i ci?;gn?, wy;ci?;gn?? nie mog?..

Za;wo;?a? wnu;czek szcze;niacz;ka Mrucz;ka,  
Przy;le;cia? Mru;czek i ci?;gnie wnucz;ka!  
Mruc;zek za wnucz;ka,  
Wnu;czek za bab;ci?,  
Bab;cia za dziad;ka,  
Dzia;dek za rzep;k?,  
Oj, przy;da?;by si? kto? na przy;czep;k?!  
Poc? si?, sapi?, st?;ka;j? srogo,  
Ci?;gn? i ci?;gn?, wy;ci?;gn?? nie mog?!

Na kurk? czy;ha? kotek w ukry;ciu,  
Za;szcze;ka? Mru;czek : äPoms? nam, Kiciu!ö  
Kicia za Mrucz;ka,  
Mruc;zek za wnucz;ka,  
Wnu;czek za bab;ci?,  
Bab;cia za dziad;ka,  
Dzia;dek za rzep;k?,  
Oj, przy;da?;by si? kto? na przy;czep;k?!  
Poc? si?, sapi?, st?;ka;j? srogo,  
Ci?;gn? i ci?;gn?, wy;ci?;gn?? nie mog?!

Wi?c wo?a Kicia kurk? z po;dwsr;ka,  
Wnet przy;le;cia;?a us?u?;na kurka.  
Kurka za Kici?,  
Kicia za Mrucz;ka,  
Mruc;zek za wnucz;ka,  
Wnu;czek za bab;ci?,  
Bab;cia za dziad;ka,  
Dzia;dek za rzep;k?,  
Oj, przy;da?;by si? kto? na przy;czep;k?!  
Poc? si?, sapi?, st?;ka;j? srogo,  
Ci?;gn? i ci?;gn?, wy;ci?;gn?? nie mog?!



```
Bits ASCII: 31048  
Bits Huffmana: 15974  
Compression ratio: 1.94366
```

## Wnioski

Głównymi wadami są to, że program działa tylko na tekście wpisanym ręcznie w kodzie, nie odczytuje danych z pliku ani od użytkownika, nie zapisuje zakodowanego wyniku do pliku oraz nie obsługuje przypadku pustego tekstu lub tekstu z jednym znakiem, co ogranicza praktyczne zastosowanie programu.