Sprawozdanie 3

Józef Piechaczek

2019-04-20

1 Zadanie 1

Celem zadania 1 jest napisanie programu ramkującego zgodnie z zasadą "rozpychania bitów" oraz weryfikującego poprawność ramki metodą CRC. Program pobiera z pliku testowego ciąg złożony ze znaków '0' i '1', który symuluje strumień bitów i tworzy na jego podstawie ramki, które następnie zapisuje do pliku wyjściowego. Nazwy plików wejściowych i wyjściowych przyjmowane są jako argumenty wywołania programu.

Algorytm zamieniania strumienia bitów na ramki:

1. Pobranie fragmentu strumienia bitów
2. Obliczenie sumy kontrolnej CRC
3. Konkatenacja fragmentu i sumy CRC
4. Dodanie wartości "0" po każdym ciągu "011111"
5. Dodanie sekwencji "01111110" na początku i końcu ramki

Program obsługuje również procedurę odwrotną, czyli zamianę ramek na tekst wyjściowy.

Algorytm zamieniania ramek na strumień bitów:

1. Pobranie kolejnej ramki
2. Odnalezienie początku i końca wiadomości na podstawie ułożenia sekwencji "01111110"
3. Usunięcie wartości "0" po każdym ciągu "011111"
4. Oddzielenie pola CRC od wiadomości
5. Obliczenie sumy CRC dla uzyskanej wiadomości i porównanie z podanym CRC
6. Zwrócenie wiadomości, jeśli sumy CRC się zgadzają, lub błędu, dla różnych sum kontrolnych

1.1 Przykład 1

Umieśćmy w pliku wejściowym następujący ciąg bitów:

011111101010000111110111

Po zamienieniu wiadomości na ramki uzyskujemy następujący tekst

01111110 011111**0**10101000011111**0**0111 1000111010001010000010011010010 1 01111110

gdzie

- pierwsza i ostatnia część to sekwencje oddzielające
- druga część to wiadomość z zaznaczonymi dodatkowymi bitami
- trzecia część to kod CRC

Powyższy tekst zostaje umieszczony w pliku wynikowym.

W celu przetestowanie poprawności ramkowania posłużę się metodą odwrotną - zamieniającą ramki z poprzedniego pliku wyjściowego na źródłowy strumień bitów i zapisującą do poprzedniego pliku wejściowego. Wartość po wykonaniu operacji odwrotnej jest identyczna do wartości sprzed ramkowania.

1.2 Przykład 2

W przykładzie drugim posłużę się ciągiem bitów z poprzedniego podpunktu. Uruchamiam program tworzący ramki na postawie strumienia bitów. W pliku wyjściowym umieszczona zostaje następująca ramka z wiadomością:

01111110 01111101010100001111100111 10001110100010100000100110100101 01111110

Zamieniam kilka bitów w części zawierającej sumy kontrolne CRC:

01111110 01111101010100001111100111 10001110100010**11111**100110100101 01111110

Podczas uruchamiania programu uzyskuję następujący rezultat:

```
Exception in thread "main" exceptions.IncorrectCRCException
        at crc.Decoder.fromFrame(Decoder.kt:19)
        at crc.Decoder.decodeMessage(Decoder.kt:25)
        at Zad1Kt.decode(Zad1.kt:33)
        at Zad1Kt.main(Zad1.kt:38)
```

Wyjątek oznacza że sumy kontrolne nie są równe, co informuje o niezamierzonej zmianie bitów.

2 Zadanie 2

Celem zadania 2 jest napisanie symulacji metody dostępu do medium transmisyjnego CSMA/CD. Łącze realizowane jest za pomocą tablicy. Czas działania symulatora podzielony jest na interwały o określonej długości. W danym interwale ramki znajdujące się na łączu zostają przesunięte, a każda ze stacji podejmuje decyzję, jakie działanie powinna podjąć.

W danym interwale stacje sprawdzają, czy jest możliwość wysyłania ramek - jeśli łącze w danym miejscu jest wolne, stacje z określonym prawdopodobieństwem mogą zacząć transmisję, jeśli łącze jest zajęte, stacje milczą.

Możliwa jest sytuacja, gdy dwie lub więcej stacji zaczną transmisję jednocześnie. Wtedy każda ze stacji, po zauważeniu nieprawidłowych pakietów, wstrzymuje transmisję na okres czasu propagacji jednego pakietu. Po upływie tego czasu każda ze stacji losuje szczeliny czasowe zgodnie ze wzorem

$$T = \begin{cases} t * 2^k & \text{, gdy } 1 \leqslant k \leqslant 10 \\ t * 2^{10} & \text{, gdy } 10 \leqslant k \leqslant 16 \end{cases}$$

gdzie

- k - numer kolizji
- t - czas propagacji pakietu