Technologie Sieciowe
Sprawozdanie do listy 3

Szymon Wojtaszek 236592

May 2018

Zadanie 1

Wymagania

Napisz program ramkujący zgodnie z zasadą "rozpychania bitów" (podaną na wykładzie), oraz weryfikujący poprawność ramki metodą CRC . Program ma odczytywać pewien źródłowy plik tekstowy 'Z' zawierający dowolny ciąg złożony ze znaków '0' i '1' (symulujący strumień bitów) i zapisywać ramkami odpowiednio sformatowany ciąg do innego pliku tekstowego 'W'. Program powinien obliczać i wstawiać do ramki pola kontrolne CRC - formatowane za pomocą ciągów złożonych ze znaków '0' i '1'. Napisz program, realizujący procedure odwrotną, tzn. który odczytuje plik.

Realizacja

Za główna funkcjonalność programu odpowiada klasa **PackageConverter**, której interface wygląda następująco:

```
1 public:
2     PackageConverter(const string & crc_v) : crc_value(crc_v){}
3     const string isolate(const string & package_str) const;
4     const string package(const string & str) const;
```

Konstruktor pobiera wartosc string, ktora staje sie wielomianem CRC, dla algorytmu o tej samej nazwie

Funkcja **string package(const string &str)** pobiera wiadomość, a następnie koduje ją w nowo utworzonym pakiecie. Używa ona dwóch specjalnych znaków ASCII **SOH** (ang. Start Of Header) oraz **EOT** (ang. End Of Transmission), które umieszcza na początku oraz na końcu pakietu. W celu eliminacji błędów spowodowanych wystąpieniem w wiadomości ciągów bitów odpowiadających zarezerwowanym znakom funkcja korzysta z techniki "nadziewania bitów" polegającej na szukaniu w wiadomości zarezerwowanych znaków i podstawieniu w ich miejsce specjalnej sekwencji zastępczej. Do tego celu został wyróżniony specjalny znak **ESC**, służący do wskazania, że następną wartość trzeba zamienić na znak specjalny. Program używa następujących podstawień:

Znak specjalny	Znak zastępczy
SOH	A
EOT	B
ESC	C

Tabela 1: Tablica odwzorowania znaków

Niemniej jednak w celu lepszej czytelności programu wartości te zostały ukryte pod stałymi symbolicznymi np. **INSTEAD_OF_SOH**. Na końcu funkcji obliczamy wartość crc i dodając ją do ramki. Kod całej funkcji wygląda następująco:

```
1  const std::string PackageConverter::package(const std::string
   ↪  &str) const {
2      const int ASCII_CHAR_SIZE = 8;
3
4      string frame;
5      string temp;
6      frame+=SOH;
7      int i = 0;
8      while(i < str.size()){
9          temp = str.substr(i, ASCII_CHAR_SIZE);
10         if(temp == SOH){
11             frame += ESC;
```

```
            frame += INSTEAD_OF_SOH;
            i += ASCII_CHAR_SIZE;
        } else if(temp == EOT){
            frame += ESC;
            frame += INSTEAD_OF_EOT;
            i += ASCII_CHAR_SIZE;
        } else if(temp == ESC){
            frame += ESC;
            frame += INSTEAD_OF_ESC;
            i += ASCII_CHAR_SIZE;
        } else {
            frame += str.at(i);
            i++;
        }

    }
    frame += EOT;
    frame += crc(str);

    return frame;
}
```

Funkcja **string isolate(const string &packaged_str)** działa w sposób analogiczny. Najpierw szuka w otrzymanej wiadomości znaku początku ramki, a następnie do czasu wystąpienia **EOT** idąc po wiadomości szuka znaków **ESC**, a po natrafieniu zamienia sekwencje zastępczą na odpowiadający jej ciąg bitów. Funkcja przechwytuje dwa rodzaje błędów:

- **Zły format ramki**, jeśli w ramce do przetworzenia brakuje znaku **SOH** lub **EOT**.
- **Błędny klucz crc**, w przypadku braku zgodności klucza crc odczytanego z ramki z obliczonym z wyodrębnionej wiadomości.

Cały kod funkcji **string isolate(const string &packaged_str)** wygląda następująco

```
const std::string PackageConverter::isolate(const std::string
↪  &packaged_str) const {
    const int ASCII_CHAR_SIZE = 8;
```

```
3       string str;
4       string temp;
5       int i = 0;
6       while(packaged_str.substr(i, SOH.size()) != SOH && i <
        ↪  packaged_str.size()){
7           i++;
8       }
9       i += ASCII_CHAR_SIZE;
10
11      while(packaged_str.substr(i, EOT.size()) != EOT && i <
        ↪  packaged_str.size()){
12          if(packaged_str.substr(i, ESC.size()) == ESC){
13              i += ASCII_CHAR_SIZE;
14              temp = packaged_str.substr(i, ASCII_CHAR_SIZE);
15              if(temp == INSTEAD_OF_SOH){
16                  str += SOH;
17                  i += ASCII_CHAR_SIZE;
18              } else if(temp == INSTEAD_OF_EOT){
19                  str += EOT;
20                  i += ASCII_CHAR_SIZE;
21              } else if(temp == INSTEAD_OF_ESC){
22                  str += ESC;
23                  i += ASCII_CHAR_SIZE;
24              } else {
25                  std::cerr << "Znaleziono ESC, ale nie
                    ↪  znaleziono zakodowanego znaku za nia, to
                    ↪  nie powinno sie wydarzyc" << std::endl;
26              }
27          } else{
28              str += packaged_str.at(i);
29              i++;
30          }
31      }
32      i += ASCII_CHAR_SIZE;
33
34      if (i >= packaged_str.size()){
35          std::cerr << "Zly format ramki" << std::endl;
36          return "";
```

```
37      }
38      if(packaged_str.substr(i, crc_value.size() - 1) !=
        ↪  crc(str)){
39          std::cerr << packaged_str << std::endl;
40          std::cerr << str << std::endl;
41          std::cerr << i << std::endl;
42          std::cerr << packaged_str.substr(i, 10000) <<
            ↪  std::endl;
43          std::cerr << "Bledny klucz crc!" << std::endl;
44          return "";
45      }
46
47      return str;
48  }
```

W obu funkcjach jest wykorzystywana prywatna funkcja **string crc(const string & str)** (ang. cyclic redundancy check) obliczająca sumę kontrolną pozwalającą wykryć ewentualne błędy w transmisji pakietu. Cykliczny kod nadmiarowy definiju się jako resztę z dzielenia liczby reprezentującą wiadomość przez określony dzielnik CRC, określany przy tworzeniu obiektu konwentera. Kod funkcji wygląda następująco:

```
1   const std::string PackageConverter::crc(const std::string &
    ↪  str) const {
2       string code = str;
3       for(int i = 0; i < crc_value.size() - 1; i++)
4           code += "0";
5
6       for(int i = 0; i < str.size(); i++){
7           if(code.at(i) == '0')
8               continue;
9           for(int j = 0; j < crc_value.size(); j++){
10              if(code.at(i + j) == crc_value.at(j))
11                  code.at(i + j) = '0';
12              else
13                  code.at(i + j) = '1';
14          }
15      }
16
```

```
17      return code.substr(code.size() - (crc_value.size() - 1),
        ↪  crc_value.size() - 1);
18  }
```

Wnioski z zadania pierwszego

Tworzenie symulacji pozwoliło na doświadczalne zapoznanie się z procesem formowania ramki, zapoznaniem się z jej najważniejszymi elementami, zrozumienie techniki nadziewania bitami, oraz oswojenie się z algorytmem obliczającym sumy kontrolne.

Zadanie 2

Wymagania

Napisz program (grupę programów) do symulowania ethernetowej metody dostepu do medium transmisyjnego (CSMA/CD). Wspólne łącze realizowane jest za pomocą tablicy: propagacja sygnału symulowana jest za pomoca propagacji wartości do sąsiednich komórek. Zrealizuj ćwiczenie tak, aby symulacje można było w łatwy sposób testować i aby otrzymane wyniki były łatwe w interpretacji.

Realizacja

Główny trzon programu opiera się na klasie **Host** z prostym interfejsem publicznym:

```
public:
    Host(string name_v, Cable & cable_v, int port_v);
    void propagate(const string & message);
```

Każdemu egzemplarzowi klasy **Host** nadajemy przy tworzeniu imie, referencje do obiektu Cable, oraz numer portu (miejsce w kablu), gdzie chcemy nasz klos podpiąc. Klasa Cable to jest prosty kontener zawierający tablicę znaków oraz funkcję: **void lock()** i **void unlock()** pozwalające wyeliminować błędy wielowątkowości programu.

Funkcja **void propagate(const string & message)** wysyła wiadomość do kanału komunikacyjnego zgodnie z algorytmem **CSMA/CD** (ang. Carrier Sense Multiple Access with Collision Detection). Po otrzymaniu polecenia

nadawania wiadomości funkcja cyklicznie sprawdza dostępność kanału. Kiedy w końcu kanał jest wolny zaczyna ona nadawać wiadomość sprawdzając przed początkiem każdej iteracji, czy w kanale nie doszło do kolizji. W takim przypadku funkcja przestaje nadawać i czeka losowy czas z przedziału (0, wait_time), a po nim ponownie próbuje nadać wiadomość. Zrandomizowany czas oczekiwania ma na celu wyeliminować przypadek, w którym dwa hosty po napotkaniu kolizji czekają dokładnie ten sam przedział czasu po czym znów dochodzi do kolizji.

W przypadku bardziej obciążonej sieci, aby zminimalizować zastój spowodowany częstymi kolizjami algorytm CSMA/CD wykorzystuje strategię **Binary Exponential Backoff** polegającą na podwojeniu czasu oczekiwania po każdym napotkaniu na kolizję co szybko pozwala zwiększyć przedział czasu tak, aby mogło dojść do wymiany informacji.

Kod funkcji jest następujący:

```
void Host::propagate(const string & message) {
        std::random_device random_device;
        std::default_random_engine
        ↪ random_engine(random_device());
        int wait_time = START_WAIT_GAP;
        bool jammed = true;

        while (jammed) {
            if(cable[port] == '0') {
                int distance_from_port = 0;
                while ((port - int(message.size()) +
                ↪ distance_from_port < cable.size()) ||
                       (port + int(message.size()) -
                       ↪ distance_from_port >= 0)) {

                    cable.lock();

                    //Wykrywamy kolizje
                    bool b_left = port - distance_from_port >=
                    ↪ 0 ? cable[port - distance_from_port] !=
                    ↪ '0': false;
```

```
bool b_right = port + distance_from_port <
 ↪  cable.size() ? cable[port +
 ↪  distance_from_port] != '0' : false;
if(b_left || b_right){
    if(b_left){
        for(int i = port -
         ↪  distance_from_port + 2; i <
         ↪  std::min(port +
         ↪  distance_from_port - 1,
         ↪  cable.size()); i++)
            cable[i] = '0';
        cable[port - distance_from_port] =
         ↪  '0';
    }
    if(b_right) {
        for (int i = std::max(port -
         ↪  distance_from_port + 1, 0); i <
         ↪  port + distance_from_port - 1;
         ↪  i++)
            cable[i] = '0';
        cable[port + distance_from_port] =
         ↪  '0';
    }

    std::cout << "Host: " << host_name << "
     ↪  wykryl kolizje\n";
    cable.unlock();
    std::uniform_int_distribution<int>
     ↪  uniform_dist(1, wait_time);

     ↪  std::this_thread::sleep_for(std::chrono::milliseconds(
    wait_time *= 2;
    jammed = true;

    break;
}

//Przeswamy wiadomosc o jeden element
```

```
for (int letter_of_message = 0;
 ↪  letter_of_message < message.size();
 ↪  letter_of_message++) {
    int letter_distance_from_port =
     ↪  distance_from_port -
     ↪  letter_of_message;
    if (letter_distance_from_port == 0)
        cable[port] =
         ↪  message.at(letter_of_message);

    if (letter_distance_from_port == 1) {
        if (port - 1 >= 0 && port + 1 <
         ↪  cable.size()) {
            cable[port - 1] = cable[port +
             ↪  1] = cable[port];
            cable[port] = '0';
        } else if (port - 1 >= 0)
            shift_left(port - 1);
        else if (port + 1 < cable.size())
            shift_right(port + 1);
    }

    if (letter_distance_from_port > 1) {
        if (port +
         ↪  letter_distance_from_port <
         ↪  cable.size())

             ↪  shift_right(letter_distance_from_port
             ↪  + port);
        if (port +
         ↪  letter_distance_from_port ==
         ↪  cable.size())
            cable[cable.size() - 1] = '0';

        if (port -
         ↪  letter_distance_from_port >= 0)
            shift_left(port -
             ↪  letter_distance_from_port);
```

```
                        if (port -
                        ↪  letter_distance_from_port ==
                        ↪  -1)
                            cable[0] = '0';
                    }

                }
                cable.print();
                cable.unlock();


                ↪  std::this_thread::sleep_for(std::chrono::milliseconds(2));

                distance_from_port++;
                jammed = false;
            }
        } else {

            ↪  std::this_thread::sleep_for(std::chrono::milliseconds(INTERFRA
        }
        cable.print();
    }
}
```

W celu testowania przedstawionej klasy tworzymy obiekt medium oraz trzy hosty z określonymi portami, a następnie każdemu hostowi każemy nadawać wiadomość w osobnym wątku:

```
const int CABLE_SIZE = 62;

int main(){

    Cable cable(CABLE_SIZE);
    Host host1("host1", cable, 0);
    Host host2("host2", cable, CABLE_SIZE / 2);
    Host host3("host3", cable, CABLE_SIZE - 1);

    std::thread thread1(&Host::propagate, &host1, " AA ");
```

```
    std::thread thread2(&Host::propagate, &host2, " B ");
    std::thread thread3(&Host::propagate, &host3, " XYZ ");

    thread1.join();
    thread2.join();
    thread3.join();

    return 0;
}
```

Otrzymujemy następujące wyniki:

```
/home/pan/Studia/Semestr4/sieci/lista3/zad2/cmake-build-debug/zad2
 0000000000000000000000000000000000000000000000000000000000000000
 00000000000000000000000000000 00000000000000000000000000000
 00000000000000000000000000000 00000000000000000000000000000
A 00000000000000000000000000000 00000000000000000000000000000
A 00000000000000000000000000000 B 00000000000000000000000000000
A 00000000000000000000000000000 B 00000000000000000000000000000 X
AA 00000000000000000000000000000 B 00000000000000000000000000000 X
AA 00000000000000000000000000000 B B 00000000000000000000000000000 X
AA 00000000000000000000000000000 B B 00000000000000000000000000000 XY
 AA 00000000000000000000000000000 B B 00000000000000000000000000000 XY
 AA 00000000000000000000000000000 B 0 B 00000000000000000000000000000 XY
 AA 00000000000000000000000000000 B 0 B 00000000000000000000000000000 XYZ
0 AA 00000000000000000000000000000 B 0 B 00000000000000000000000000000 XYZ
0 AA 00000000000000000000000000000 B 000 B 00000000000000000000000000000 XYZ
0 AA 00000000000000000000000000000 B 000 B 00000000000000000000000000000 XYZ
0 AA 00000000000000000000000000000 B 00000 B 00000000000000000000000000000 XYZ
00 AA 00000000000000000000000000000 B 00000 B 00000000000000000000000000000 XYZ
00 AA 00000000000000000000000000000 B 00000 B 00000000000000000000000000000 XYZ 0
00 AA 00000000000000000000000000000 B 0000000 B 00000000000000000000000000000 XYZ 0
000 AA 00000000000000000000000000000 B 0000000 B 00000000000000000000000000000 XYZ 0
000 AA 00000000000000000000000000000 B 0000000 B 00000000000000000000000000000 XYZ 00
000 AA 00000000000000000000000000000 B 0000000 B 00000000000000000000000000000 XYZ 000
0000 AA 00000000000000000000000000000 B 0000000 B 00000000000000000000000000000 XYZ 000
0000 AA 00000000000000000000000000000 B 000000000 B 000000000000000000000000000 XYZ 000
0000 AA 00000000000000000000000000000 B 000000000 B 000000000000000000000000000 XYZ 0000
0000 AA 00000000000000000000000000000 B 00000000000 B 00000000000000000000000000 XYZ 0000
```

```
00000 AA 000000000000000 B 00000000000 B 0000000000000 XYZ 0000
00000 AA 000000000000000 B 00000000000 B 000000000000 XYZ 00000
00000 AA 0000000000000 B 0000000000000 B 00000000000 XYZ 00000
000000 AA 000000000000 B 0000000000000 B 00000000000 XYZ 00000
000000 AA 000000000000 B 0000000000000 B 0000000000 XYZ 000000
000000 AA 00000000000 B 000000000000000 B 000000000 XYZ 000000
0000000 AA 0000000000 B 000000000000000 B 000000000 XYZ 000000
0000000 AA 0000000000 B 000000000000000 B 00000000 XYZ 0000000
00000000 AA 000000000 B 000000000000000 B 00000000 XYZ 0000000
00000000 AA 00000000 B 00000000000000000 B 0000000 XYZ 0000000
00000000 AA 00000000 B 00000000000000000 B 000000 XYZ 00000000
000000000 AA 0000000 B 00000000000000000 B 000000 XYZ 00000000
000000000 AA 000000 B 000000000000000000 B 00000 XYZ 00000000
000000000 AA 00000 B 0000000000000000000 B 0000 XYZ 00000000
0000000000 AA 0000 B 0000000000000000000 B 0000 XYZ 00000000
0000000000 AA 0000 B 0000000000000000000 B 000 XYZ 000000000
00000000000 AA 000 B 0000000000000000000 B 000 XYZ 000000000
00000000000 AA 00 B 00000000000000000000 B 00 XYZ 000000000
00000000000 AA 00 B 00000000000000000000 B 0 XYZ 0000000000
000000000000 AA 0 B 00000000000000000000 B 0 XYZ 0000000000
000000000000 AA  B 00000000000000000000 B  XYZ 0000000000
Host: host3 wykryl kolizje
Host: host1 wykryl kolizje
Host: host2 wykryl kolizje
0000000000000000000000000000000000000000000000000000000000000000
 0000000000000000000000000000000000000000000000000000000000000000
A 0000000000000000000000000000000000000000000000000000000000000000
AA 0000000000000000000000000000000000000000000000000000000000000000
 AA 0000000000000000000000000000000000000000000000000000000000000000
0 AA 0000000000000000000000000000000000000000000000000000000000000000
00 AA 0000000000000000000000000000000000000000000000000000000000000000
000 AA 0000000000000000000000000000000000000000000000000000000000000000
0000 AA 0000000000000000000000000000000000000000000000000000000000000000
00000 AA 0000000000000000000000000000000000000000000000000000000000000000
000000 AA 0000000000000000000000000000000000000000000000000000000000000000
0000000 AA 0000000000000000000000000000000000000000000000000000000000000000
00000000 AA 0000000000000000000000000000000000000000000000000000000000000000
000000000 AA 0000000000000000000000000000000000000000000000000000000000000000
```

0000000000 AA 0000000000000000000000000000000000000000000000000000
00000000000 AA 0000000000000000000000000000000000000000000000000000
00000000000 AA 0000000000000000000000000000000000000000000000000000
00000000000 AA 000000000000000000000000000000000000000000000000000
000000000000 AA 000000000000000000000000000000000000000000000000000
000000000000 AA 0000000000000000000000000000000000000000000000000000 X
0000000000000 AA 0000000000000000000000000000000000000000000000000000 X
0000000000000 AA 0000000000000000000000000000000000000000000000000000 XY
00000000000000 AA 0000000000000000000000000000000000000000000000000000 XY
00000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ
000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ
000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ
0000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ
0000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0
00000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0
00000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00
000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00
000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000
0000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000
0000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0000
00000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0000
00000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00000
000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00000
000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000000
0000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000000
0000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0000000
00000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0000000
00000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00000000
000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00000000
000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000000000
0000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000000000
0000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0000000000
00000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0000000000
00000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00000000000
000000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 00000000000
000000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000000000000
0000000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 000000000000
0000000000000000000000000000 AA 0000000000000000000000000000000000000000000000000000 XYZ 0000000000000

```
00000000000000000000000000000 AA 00000000000 XYZ 0000000000000
00000000000000000000000000000 AA 0000000000 XYZ 00000000000000
000000000000000000000000000000 AA 000000000 XYZ 00000000000000
000000000000000000000000000000 AA 00000000 XYZ 000000000000000
0000000000000000000000000000000 AA 0000000 XYZ 000000000000000
0000000000000000000000000000000 AA 000000 XYZ 0000000000000000
00000000000000000000000000000000 AA 00000 XYZ 0000000000000000
00000000000000000000000000000000 AA 0000 XYZ 00000000000000000
000000000000000000000000000000000 AA 000 XYZ 00000000000000000
000000000000000000000000000000000 AA 00 XYZ 000000000000000000
0000000000000000000000000000000000 AA 0 XYZ 000000000000000000
0000000000000000000000000000000000 AA  XYZ 0000000000000000000
Host: host1 wykryl kolizje
Host: host3 wykryl kolizje
00000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000 000000000000000000000000000000
000000000000000000000000000000 B 00000000000000000000000000000
00000000000000000000000000000 B B 0000000000000000000000000000
0000000000000000000000000000 B 0 B 000000000000000000000000000
000000000000000000000000000 B 000 B 00000000000000000000000000
00000000000000000000000000 B 00000 B 0000000000000000000000000
0000000000000000000000000 B 0000000 B 000000000000000000000000
000000000000000000000000 B 000000000 B 00000000000000000000000
00000000000000000000000 B 00000000000 B 0000000000000000000000
0000000000000000000000 B 0000000000000 B 000000000000000000000
000000000000000000000 B 000000000000000 B 00000000000000000000
00000000000000000000 B 00000000000000000 B 0000000000000000000
0000000000000000000 B 0000000000000000000 B 000000000000000000
000000000000000000 B 000000000000000000000 B 00000000000000000
00000000000000000 B 00000000000000000000000 B 0000000000000000
0000000000000000 B 0000000000000000000000000 B 000000000000000
000000000000000 B 000000000000000000000000000 B 00000000000000
00000000000000 B 00000000000000000000000000000 B 0000000000000
0000000000000 B 0000000000000000000000000000000 B 000000000000
000000000000 B 000000000000000000000000000000000 B 00000000000
00000000000 B 00000000000000000000000000000000000 B 0000000000
0000000000 B 0000000000000000000000000000000000000 B 000000000
000000000 B 000000000000000000000000000000000000000 B 00000000
```

00000000 B 00000000000000000000000000000000000000000000 B 0000000
0000000 B 000000000000000000000000000000000000000000000 B 000000
000000 B 0000000000000000000000000000000000000000000000 B 00000
00000 B 00000000000000000000000000000000000000000000000 B 0000
0000 B 000000000000000000000000000000000000000000000000 B 000
000 B 0000000000000000000000000000000000000000000000000 B 00
00 B 00000000000000000000000000000000000000000000000000 B 0
0 B 000000000000000000000000000000000000000000000000000 B
 B 0000000000000000000000000000000000000000000000000000000 B
B 00000000000000000000000000000000000000000000000000000000
 000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000
 000000000000000000000000000000000000000000000000000000000000
A 0000000000000000000000000000000000000000000000000000000000000
AA 000000000000000000000000000000000000000000000000000000000000
 AA 0000000000000000000000000000000000000000000000000000000000
0 AA 00000000000000000000000000000000000000000000000000000000000
00 AA 000000000000000000000000000000000000000000000000000000000
000 AA 0000000000000000000000000000000000000000000000000000000
0000 AA 00000000000000000000000000000000000000000000000000000
00000 AA 000000000000000000000000000000000000000000000000000
000000 AA 0000000000000000000000000000000000000000000000000
0000000 AA 00000000000000000000000000000000000000000000000
00000000 AA 000000000000000000000000000000000000000000000
000000000 AA 0000000000000000000000000000000000000000000
0000000000 AA 00000000000000000000000000000000000000000
00000000000 AA 000000000000000000000000000000000000000
000000000000 AA 0000000000000000000000000000000000000
0000000000000 AA 00000000000000000000000000000000000
00000000000000 AA 000000000000000000000000000000000
000000000000000 AA 0000000000000000000000000000000
0000000000000000 AA 00000000000000000000000000000
00000000000000000 AA 000000000000000000000000000
000000000000000000 AA 0000000000000000000000000
0000000000000000000 AA 00000000000000000000000
00000000000000000000 AA 000000000000000000000
000000000000000000000 AA 0000000000000000000
0000000000000000000000 AA 00000000000000000
00000000000000000000000 AA 000000000000000
000000000000000000000000 AA 0000000000000
0000000000000000000000000 AA 00000000000
00000000000000000000000000 AA 000000000
000000000000000000000000000 AA 0000000
0000000000000000000000000000 AA 00000
00000000000000000000000000000 AA 000
000000000000000000000000000000 AA 0

000000000000000000000 AA 0000000000000000000000000000000000000
0000000000000000000000 AA 000000000000000000000000000000000000
00000000000000000000000 AA 00000000000000000000000000000000000
000000000000000000000000 AA 0000000000000000000000000000000000
0000000000000000000000000 AA 000000000000000000000000000000000
00000000000000000000000000 AA 00000000000000000000000000000000
000000000000000000000000000 AA 0000000000000000000000000000000
0000000000000000000000000000 AA 000000000000000000000000000000
00000000000000000000000000000 AA 00000000000000000000000000000
000000000000000000000000000000 AA 0000000000000000000000000000
0000000000000000000000000000000 AA 000000000000000000000000000
00000000000000000000000000000000 AA 00000000000000000000000000
000000000000000000000000000000000 AA 0000000000000000000000000
0000000000000000000000000000000000 AA 000000000000000000000000
00000000000000000000000000000000000 AA 00000000000000000000000
000000000000000000000000000000000000 AA 0000000000000000000000
0000000000000000000000000000000000000 AA 000000000000000000000
00000000000000000000000000000000000000 AA 00000000000000000000
000000000000000000000000000000000000000 AA 0000000000000000000
0000000000000000000000000000000000000000 AA 000000000000000000
00000000000000000000000000000000000000000 AA 00000000000000000
000000000000000000000000000000000000000000 AA 0000000000000000
0000000000000000000000000000000000000000000 AA 000000000000000
00000000000000000000000000000000000000000000 AA 00000000000000
000000000000000000000000000000000000000000000 AA 0000000000000
0000000000000000000000000000000000000000000000 AA 000000000000
00000000000000000000000000000000000000000000000 AA 00000000000
000000000000000000000000000000000000000000000000 AA 0000000000
0000000000000000000000000000000000000000000000000 AA 000000000
00000000000000000000000000000000000000000000000000 AA 00000000
000000000000000000000000000000000000000000000000000 AA 0000000
0000000000000000000000000000000000000000000000000000 AA 000000
00000000000000000000000000000000000000000000000000000 AA 00000
000000000000000000000000000000000000000000000000000000 AA 0000
0000000000000000000000000000000000000000000000000000000 AA 000
00000000000000000000000000000000000000000000000000000000 AA 00
000000000000000000000000000000000000000000000000000000000 AA 0
0000000000000000000000000000000000000000000000000000000000 AA

0000000000000000000000000000000000000000000000000000000000000000 AA
0000000000000000000000000000000000000000000000000000000000000000 A
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 X
0000000000000000000000000000000000000000000000000000000000000000 XY
0000000000000000000000000000000000000000000000000000000000000000 XYZ
0000000000000000000000000000000000000000000000000000000000000000 XYZ
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 00000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000 XYZ 0000000000000000000000000000000000000000000000000000000000000000

```
00000000000000000000000000000 XYZ 000000000000000000000000000
0000000000000000000000000000 XYZ 0000000000000000000000000000
000000000000000000000000000 XYZ 00000000000000000000000000000
00000000000000000000000000 XYZ 000000000000000000000000000000
0000000000000000000000000 XYZ 0000000000000000000000000000000
000000000000000000000000 XYZ 00000000000000000000000000000000
00000000000000000000000 XYZ 000000000000000000000000000000000
0000000000000000000000 XYZ 0000000000000000000000000000000000
000000000000000000000 XYZ 00000000000000000000000000000000000
00000000000000000000 XYZ 000000000000000000000000000000000000
0000000000000000000 XYZ 0000000000000000000000000000000000000
000000000000000000 XYZ 00000000000000000000000000000000000000
00000000000000000 XYZ 000000000000000000000000000000000000000
0000000000000000 XYZ 0000000000000000000000000000000000000000
000000000000000 XYZ 00000000000000000000000000000000000000000
00000000000000 XYZ 000000000000000000000000000000000000000000
0000000000000 XYZ 0000000000000000000000000000000000000000000
000000000000 XYZ 00000000000000000000000000000000000000000000
00000000000 XYZ 000000000000000000000000000000000000000000000
0000000000 XYZ 0000000000000000000000000000000000000000000000
000000000 XYZ 00000000000000000000000000000000000000000000000
00000000 XYZ 000000000000000000000000000000000000000000000000
0000000 XYZ 0000000000000000000000000000000000000000000000000
000000 XYZ 00000000000000000000000000000000000000000000000000
00000 XYZ 000000000000000000000000000000000000000000000000000
0000 XYZ 0000000000000000000000000000000000000000000000000000
000 XYZ 00000000000000000000000000000000000000000000000000000
00 XYZ 000000000000000000000000000000000000000000000000000000
0 XYZ 0000000000000000000000000000000000000000000000000000000
 XYZ 00000000000000000000000000000000000000000000000000000000
XYZ 000000000000000000000000000000000000000000000000000000000
YZ 0000000000000000000000000000000000000000000000000000000000
Z 00000000000000000000000000000000000000000000000000000000000
 000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000

Process finished with exit code 0
```

Wnioski

Dzięki wykonanej symulacji zaobserwowaliśmy proces rozchodzenia się wiadomości nadawanych przez różne urządzenia sieciowe przez jedno medium. Zbadaliśmy działanie algorytmu CSMA/CD i pokazaliśmy, że dzięki podwajaniu czasu oczekiwania pozwala on przy niewielkiej ilości prób uniknąć dalszych kolizji.