

# Sieci Komputerowe - ULTIMATE

SonMati

3 lutego 2015

# Protokół komunikacyjny

Jest to mechanizm, który służy do wymiany informacji pomiędzy dwiema stacjami, jednostkami cyfrowymi (*DD - digital device*).

## 1 Skład protokołu

- Synchronizacja kooperacji
- Role obu obojga (*parties*)
  - Peer to Peer (P2P)
  - Master-Slave
- *Framing* (Ramkowanie, format przesyłanych danych)
- Poprawa błędów

## 2 Enkapsulacja

### 2.1 5 warstw

Aplikacja
Transport
Sieć
Link access
Fizyczna warstwa

### 2.2 3-poziomowa architektura komunikacji

- File transfer ↔ File transfer (Transport, Aplikacja)
- Transport ↔ Transport (Sieć)
  - Transport właściwy
  - Sieć (Network)
- Network Access ↔ Communication network ↔ Network Access (Link access)
  - Logiczne połączenie (Link Access)
  - Fizyczne połączenie (Physical Access)

## 3 Service Access Point (SAP)

SAP jest wykorzystywany w sieciach typu OSI (*Open Systems Interconnection*), jest to etykietę identyfikującą dla punktów końcowych sieci. Innymi słowy, oznacza ich lokalizację w sieci.

Nagłówek	Host	Punkt przeznaczenia	Dane	Kontrola błędów
----------	------	---------------------	------	-----------------

Aplikacja	Dane
Transport	Punkt przeznaczenia + Dane
Sieć	Host + Punkt przeznaczenia + Dane
Link access	Nagłówek + Host + Punkt przeznaczenia + Dane + Kontrola błędów

### 3.1 Format

Kolejność wykonywania wiadomości w protokole: Przykład sieci:



Rysunek 1.24. Hosty, routery i przełączniki warstwy łączą danych.

Każdy z tych węzłów posiada inny zestaw warstw, które są odzwierciedleniem ich różnego przeznaczenia

## Model ISO / OSI

### 1 7-warstwowy model

Jest to tzw. **stos protokołów**, zaprezentowany od "góry do dołu". Każda warstwa świadczy pewne

Warstwa aplikacyjna
Warstwa prezentacyjna
Warstwa sesji
Warstwa transportowa
Warstwa sieci
Warstwa łączą danych
Warstwa fizyczna

usługi warstwie znajdującej się ponad nią. Innymi słowy, warstwa korzysta z usług tej bezpośrednio poniżej.

- **Warstwa aplikacyjna**

- Zapewnia dostęp do środowiska OSI dla użytkownika
- Udostępnia usługi informacyjne
- Pakiety w tej warstwie nazywamy **komunikatami**

- **Warstwa prezentacyjna**

- Zapewnia niezależność procesom aplikacji od różnic w prezentacji danych (składnia), czyli spełnia rolę tłumacza.
- Jej zadaniem jest świadczenie usług, które umożliwiają komunikującym się aplikacjom zrozumienie wymienianych danych, m.in. kompresja, opis i szyfrowanie danych
- Na tym poziomie następuje enkrypcja (szyfrowanie) danych

- **Warstwa sesji**

- Zapewnia kontrolę struktury dla komunikacji między aplikacjami
- Ustanawia, zarządza i kończy połączenie (sesję) pomiędzy współpracującymi aplikacjami
- Służy do wyznaczania ram czasowych wymiany danych i synchronizowania tego procesu
- Umożliwia tworzenie punktów kontrolnych i systemów przywracania.

- **Warstwa transportowa**

- Zapewnia niezawodny, przejrzysty transfer danych pomiędzy punktami końcowymi
- Zapewnia "od końca do końca" poprawę błędów i sterowanie przepływem.
- Pakiety w tej warstwie nazywamy **segmentami**.
- Przykłady: TCP, UDP

- **Warstwa sieci**

- Odpowiada za ustawienie, utrzymanie i zakończenie połączenia
- Protokół warstwy transportowej hosta źródłowego przekazuje segment i adres docelowy warstwie sieci
- Umożliwia dostarczenie segmentu do warstwy transportowej docelowego hosta
- Pakiety w tej warstwie nazywamy **datagramami**.

- **Warstwa łącza danych**

- Zapewnia niezawodny transfer informacji przez fizyczne łącze
- Wysyła bloki (*frames*) z niezbędną synchronizacją, kontrolą błędów i kontrolą / sterowaniem przepływem
- Przesyła datagram za pośrednictwem serii routerów
- Pakiety w tej warstwie nazywamy **ramkami**

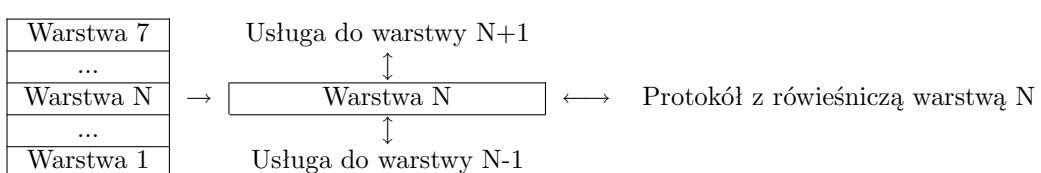
- **Warstwa fizyczna**

- Dotyczy transmisji niestrukturyzowanego strumienia bitów na fizycznym medium
- Dotyczy właściwości mechanicznych, elektrycznych, funkcjonalnych i proceduralnych
- Jej zadaniem jest przesyłanie poszczególnych bitów ramki pomiędzy dwoma węzłami

W zwykłym 5-warstwowym modelu odpowiadające mu warstwy spełniają prawie że te same role.

## 2 Architektura OSI jako Framework

**Wspólny język:** ASN-1 (*Abstract Syntax Notation-One*).  
Model ISO / OSI wykorzystywany jest w technologiach:



- TCP/IP

- LAN

- HTTP

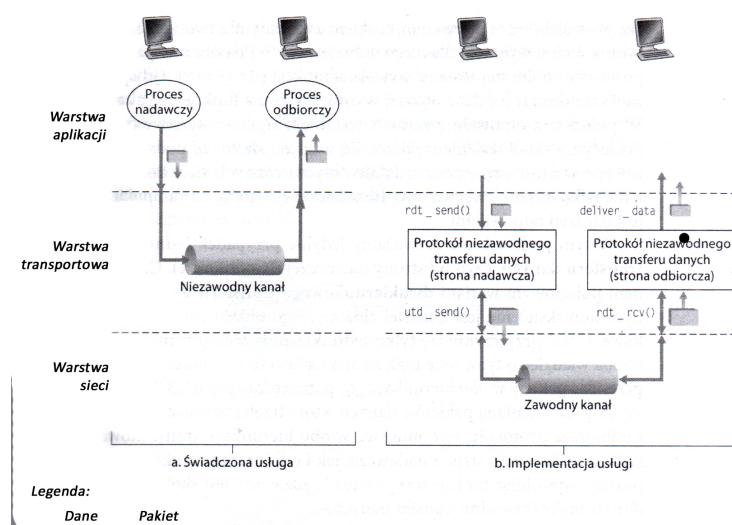
- SECURITY

## Transfer danych

Interesuje nas model **niezawodnego** transferu danych (*reliable data transfer*). Odgrywa on kluczową rolę w pracy sieci, wykorzystywany jest w warstwie transportowej, łącza danych i aplikacji.

Niezawodny kanał zapewnia, że żadne dane nie są uszkadzane, gubione oraz są dostarczane w tej samej kolejności w jakiej zostaną wysłane. Taki transport danych zapewnia np. protokół TCP.

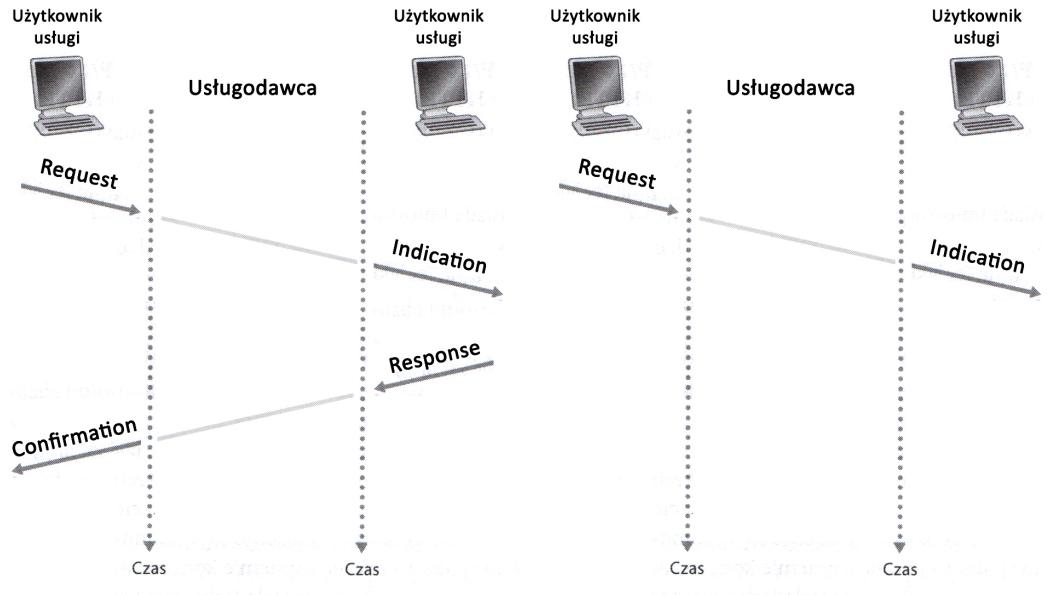
Idea jest prosta, natomiast trudna i złożona jest **implementacja** takiego protokołu.



# 1 Diagramy sekwencji czasowej dla prostych usług

## 1.1 Potwierdzona / niepotwierdzona usługa

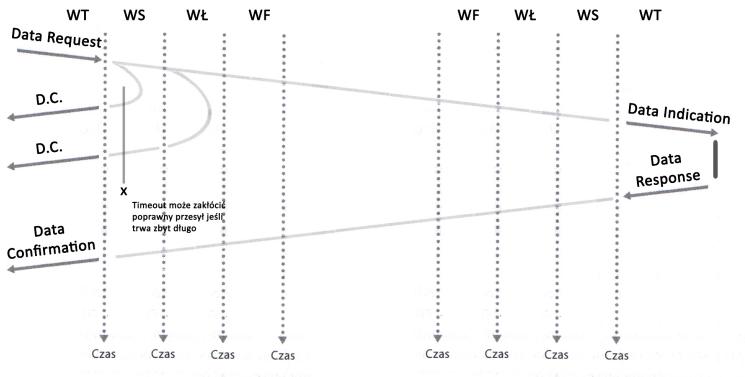
Prosty transport danych z punktu widzenia warstwy aplikacji.



Widzimy tutaj **jednokierunkowy transfer danych** (*half-duplex transfer*) - w danej chwili możliwe jest wyłącznie przesyłanie z strony nadawczej do odbiorczej.

## 1.2 Wersja z wcześniejszym powrotem

Pelny dwukierunkowy transfer (*Full two-way service*)



Nie jest wiele bardziej złożony od jednokierunkowego

# 2 Protokoły

Wykorzystujemy je do transferu danych pomiędzy warstwami i dzielimy na:

- **Znakowe** - przesyłane są pakiety danych
- **Binarne** - przesyłany jest bit po bicie

# Binarna komunikacja synchroniczna (BSC)

Jest to odmiana protokołu znakowego, który służy do implementacji niezawodnego transferu danych z obsługą błędów występujących w kanale zawodnym.

Protokół wykorzystuje mechanizm potwierdzeń.

- **Pozytywnych** - poprawna transmisja
- **Negatywnych** - prośba o powtóżenie

Aby móc obsłużyć błędny transfer protokół musi mieć zaimplementowane dodatkowe funkcje:

- **Detekcja błędów** - umożliwienie sprawdzenia u odbiorcy czy wystąpiły w błędne bity.
- **Interakcje ze strony odbiorcy** - w przypadku błędnego pakietu odbiorca musi skierować odzew do nadawcy. przykładem są właśnie pozytywne i negatywne potwierdzenia.
- **Retransmisja** - błędny transfer jest ponownie transmitowany.

## 1 Znaki kontrolne

Do realizacji zadań stawianych protokołowi BSC wykorzystano poniższą listę znaków kontrolnych:

- **STX** - Start of Text (początek tekstu)
- **SOH** - Start of Header (początek nagłówka)
- **ENQ** - Enquiry (zapytanie, np. do żądania odpowiedzi)
- **ETX** - End of Text (koniec tekstu)
- **EOT** - End of Transmission (koniec transmisji)
- **ACK** - Acknowledgement (potwierdzenie pozytywne)
- **NAK** - Negative Acknowledgement (potwierdzenie negatywne)]
- **SYN** - Synchronous Idle (czekanie, przed innymi 2 razy)
- **DLE** - Data Link Escape ()
- **ETB**

Najważniejszymi znakami kontrolnymi są ACK oraz NAK - nasze algorytmy muszą tak kontrolować te komendy, by przesył był poprawny i jak najbardziej efektywny.

## 2 Format danych

SYN	SYN	SOH	Nagłówek	STX	Tekst	ETX	BCC
-----	-----	-----	----------	-----	-------	-----	-----

## 3 Protokół komunikacji

Protokół BSC oparty jest o definicje automatów stanów skończonych (*Finite-State Machine* - FSM) dla strony nadawczej i odbiorczej. Warto zauważyć, że dla obu stron istnieją *niezależne* systemy FSM.

## 4 Schematy żądań

### 4.1 Odbioru



W sekwencji wywołania wybierany jest terminal, z którym zostanie nawiązane połączenie.

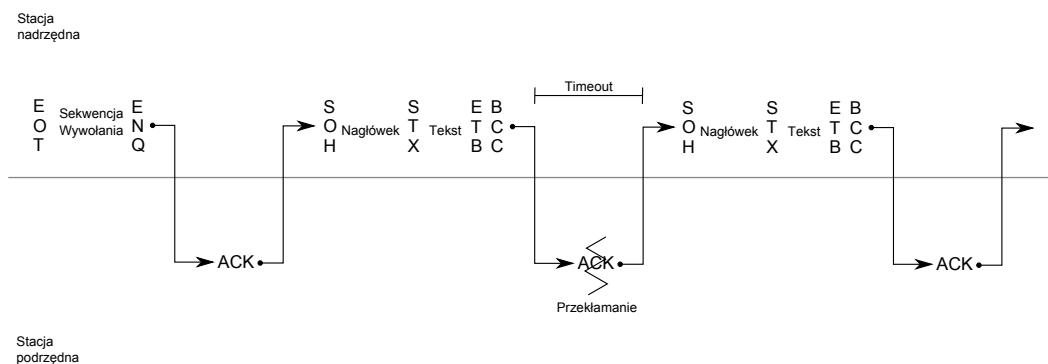
### 4.2 Nadania



### 4.3 Charakterystyka powyższych rozwiązań

- SYN SYN dajemy gdy następuje zmiana kierunku transmisji (SOH, EOT, ACK, NAK)
- Mała kontrola poprawności. Tylko ramka (SOH ... BCC) jest jakoś chroniona. Pozostałe pakiety mogą zostać przekłamane i nadawca w żaden sposób nie dowie się czy odbiorca otrzymał dobre dane. Zatem dodajemy licznik czasu (timeout), dzięki któremu może wykryć błędy jak brak potwierdzenia odbioru.

### 4.4 Z timeout



#### 4.4.1 Wada rozwiązańia

Stacja podrzędna odbiera 2 bloki tekstu, w tym jeden zduplikowany, ponieważ nie została poinformowana przez timeout o przekłamaniu. Odbiorca nie jest w stanie stwierdzić, czy pakiet jest duplikatem czy nową informacją.

### 4.5 Numeracja potwierdzeń

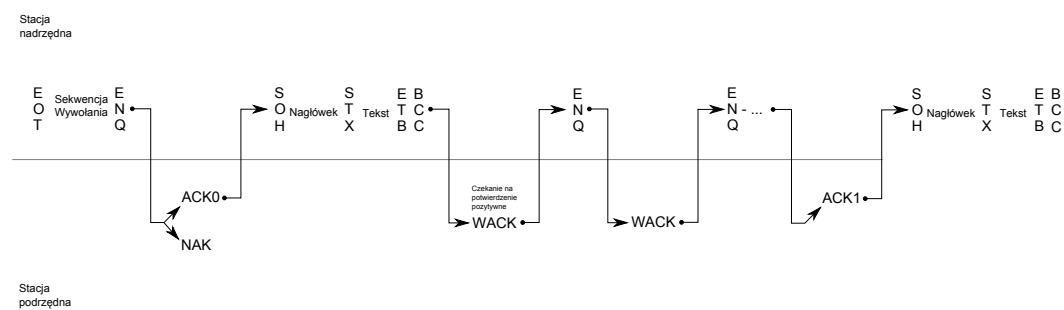
Rozwiązywanie problemu duplikatów: wprowadzenie numeru sekwencyjnego. 1-bit, zwiększający się zgodnie z arytmetyką modulo 2, umożliwia stwierdzenie czy pakiet jest wysyłany ponownie lub nie. Może być dodany do ACK lub NAK.



#### 4.5.1 Wada rozwiązańia

Jeżeli stacja nadzorująca jest zajęta, to stacja podzialek tego nie wie. To źle, bo stacja podzialek zawsze wysyła jakiś sygnał (ACK / NAK).

### 4.6 Sterowanie przepływem



**WACK** - sygnał "czekaj na potwierdzenie". Używany gdy stacja podzialek potrzebuje czasu na przetworzenie danych.

#### 4.6.1 Wada rozwiązańia

Wadą całego BSC jest brak możliwości korzystanie ze wszystkich znaków przy nagłówku i tekście, bo część z nich jest znakami kontrolnymi. Powstaje limit kombinacji bitów - brak przezroczystości. Skutkuje to tym, że nie można np. wysłać grafiki.

## 5 Przezroczysty blok w BSC



Ten umożliwia przesyłanie grafiki.

Jeśli w nagłówku lub tekście znajduje się znak DLE należy do powtórzyć.

## 6 Przebiegi czasowe

- SN: sekwencja wywołania
  - SP: Np. SYN SYN
  - SN: Blok
  - SP: 4B (SYN SYN DLE0) - czas marnowany
  - SN: znów coś

# Protokół o organizacji bitowej SDLC (Synchronous Data Link Control)

Protokół SDLC ma za zadanie rozwiązać problem wydajnościowy związany z BSC. Protokół BSC, który zdefiniowaliśmy pod sam koniec, jest protokołem zatrzymania i czekania. Sprawia, że istnieje zagrożenie nieefektywnego wykorzystania łącza - nadawca może wysyłać znacznie mniej danych w jednostce czasu niż jest to możliwe (nawet tylko ułamek procenta).

Rozwiązaniem jest zastosowanie mechanizmu **potokowego** - polega on na wysyłaniu wielu pakietów bez oczekiwania na potwierdzenie. Np. wysyłając 3 pakiety przed uzyskaniem potwierdzenia może przyspieszyć transfer 3-krotnie.

Zastosowanie tego rozwiązania niesie ze sobą konsekwencje:

- Zwiększenie zakresu numerów sekwencyjnych (każdy musi mieć swój unikatowy numer, a wysyłamy ich więcej)
  - Strona nadawcza i odbiorcza protokołu może być zmuszona do buforowania więcej niż jednego pakietu.
  - Nowe metody usuwania błędów:
    - Go-Back-N
    - Powtarzanie selektywne

# 1 Łącze, ramki i flagi

## 1.1 Format przesyłanych danych

- **Łączem** przesyłamy ciągły strumień bitów, w nim są separatory.
- Separatory to niepowtarzalne kombinacje, które nazywamy **flagami**.
- Po fladze jest **ramka** i znów flaga. Można powiedzieć, że każda ramka jest otoczona dwiema flagami.

## 1.2 Flaga

Flaga ma wartość 01111110 (inaczej 0x7E). Pojawia się problem gdy w pakiecie danych jest coś podobnego. Rozwiązanie: algorytm szpikowania zerami. Pomiędzy 5tą i 6tą jedynką wstawia się techniczne zero (011111010). przy odbiorze drugiej flagi się je kasuje.

## 1.3 Format ramki

FLAGA (1B)	Pole adresu (1B)	Pole sterujące (1B)	Dane	CRC (2B)	FLAGA (1B)
------------	------------------	---------------------	------	----------	------------

## 1.4 Adresowanie

Adres zawsze dotyczy terminala (odbiorcy lub nadawcy).

ADR = 0xFF —> transmisja rozgłoszeniowa.

### 1.4.1 Pole sterujące

- Typ I (informacyjny)  
P/F - flaga sterująca kierunkiem transmisji. Jeśli 1 to po zakończeniu odbioru następuje zmiana

1	3	1	3
0	N(S)	P/F	N(R)

kierunku.

- Typ S (sterujący numerowany)

1 0	$S_1 S_0$	P/F	N(R)
-----	-----------	-----	------

- Typ U (sterujący nienumerowany)

W pustych polach można przesyłać polecenia sterujące. Typ wykorzystywany do zamykania i otwierania połączenia oraz do stanów awaryjnych.

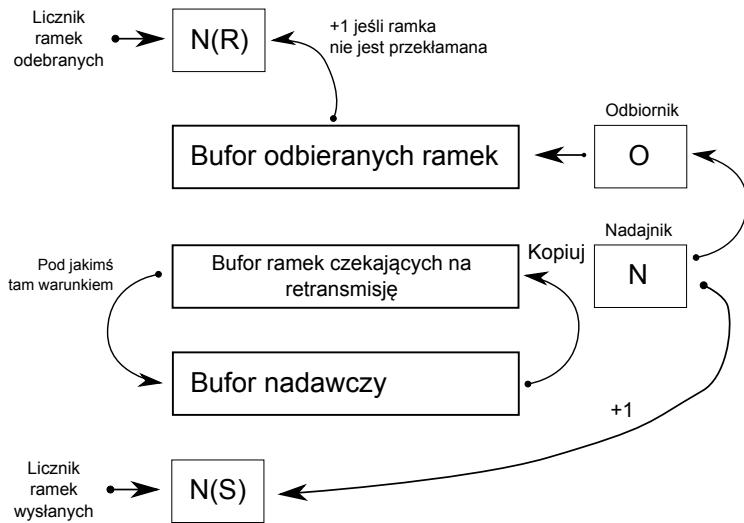
- Pierwsza ramka z SN mówi według jakiego algorytmu będzie się odbywał przesył.

- Nowsze tryby przesyłu (z większymi licznikami):

- NRM - podobne do BSC, wykorzystany typ U. Tryb normalnej odpowiedzi.
- ARM - obie stacje niezależne, bez P/F. Równoległe asynchroniczne przesyłanie.
- ABM - przesył równoczesny, usunięcie stacji nadzorujących.
- W przypadku użyciu wersji z 7-bitowym licznikiem istnieją jeszcze 3 tryby: SNRM, SARM, SABM (wszystkie z opcjonalnym E)



## 2 Adapter komunikacyjny



## 3 Komendy sterujące

- UA - nienumerowane potwierdzenie
- UI - nienumerowana informacja
- CMDR - odrzucenie polecenia (brak zgodny na dany typ, inaczej: Slave nie rozpoznaje komendy)
- FRMR - odrzucenie ramki niepasującej w ustalony schemat
- RD (*remote disconnect*), DISC, DM (*disconnect mode*) - zamykają połączenie
- XID
- Komendy numerowane
  - RR N(R) - *ready*
  - RNR N(R) - *not ready*
  - REJ N(R) - *rejection*  
Np. REJ, 1 - żądanie retransmisji ramek **od numeru 1**
  - SREJ N(R) - *selective rejection* - żądanie retransmisji wskazanej ramki. Break retransmisji grupowej.  
Np. SREJ, 1 - żądanie retransmisji **tylko** ramki nr 1.

## 4 Połączenia w SDLC

### 4.1 Normalny tryb odpowiedzi

Wykorzystuje algorytm **Go-Back-N** - "cofnij się o  $N$ ". Polega na prostej zasadzie: protokół może wysyłać wiele pakietów bez oczekiwania na potwierdzenie, jednak liczba niepotwierdzonych pakietów transferowanych w potoku nie może przekroczyć pewnej określonej liczby  $N$ . Ilustracja problemu:



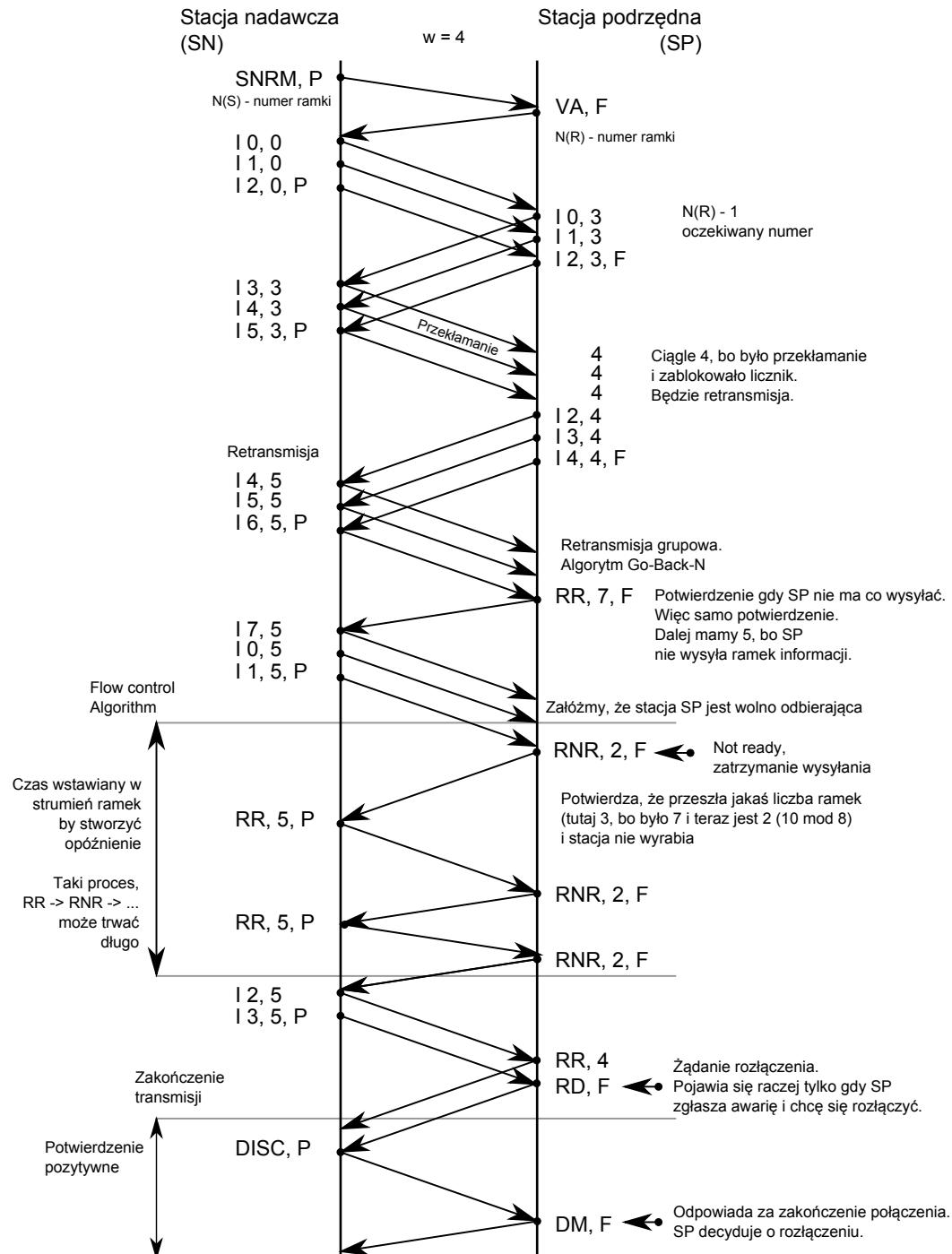
Rysunek 3.19. Numery sekwencyjne protokołu GBN z punktu widzenia nadawcy

Zakres numerów sekwencyjnych dopuszczalnych w przypadku przesyłanych pakietów, dla których nie otrzymano potwierdzenia, wyróżniono jako **okno** o rozmiarze  $N$ . W trakcie działania okno jest przesuwane wzduł zakresu numerów sekwencyjnych. Dlatego  $N$  nazywamy *rozmiarem okna*, a sam protokół GNB *protokołem przesuwnego okna*.

W protokole występują **zdarzenia**, które należy rozpoznać i obsłużyć:

- **Wywołanie z wyższej warstwy** - gdy z niej otrzymujemy pakiet, nadawca sprawdza czy okno jest pełne
  - Jeśli nie, pakiet zostaje utworzony i wysłany, a zmienne odpowiednio uaktualnione.
  - Jeśli tak, nadawca odsyła dane do warstwy wyższej, co oznacza zapelnienie okna, i wysyła dane później.
- **Odebranie potwierdzenia ACK** - powiązane z pakietem posiadającym numer sekwencyjny  $n$  zostanie uwzględnione w **potwierdzeniu skumulowanym**, które wskazuje, że wszystkie pakiety mające numery sekwencyjne aż do numeru  $n$  włacznie zostały poprawnie przekazane odbiorcy.
- **Zdarzenie upłynięcia czasu** - związane z ultraconymi lub znacznie opóźnionymi pakietami. Zegar jest używany w celu wyeliminowania ultraconych pakietów lub tych, dla których nie otrzymano potwierdzenia. Jeśli upłynie określony czas, nadawca ponownie wyśle *wszystkie* pakiety, dla których nie uzyskano potwierdzenia.

Przykład:



Optymalną wielkością okna jest 4. Stacja może wysłać tylko 4 ramki naraz, a potem musi czekać na potwierdzenie.

#### 4.1.1 Timeout

Dotyczy każdej ramki



#### 4.1.2 Rodzaje potwierdzeń

- Pozytywne
- Negatywne przez timeout

### 4.2 Tryb asynchronicznej odpowiedzi (ARM)

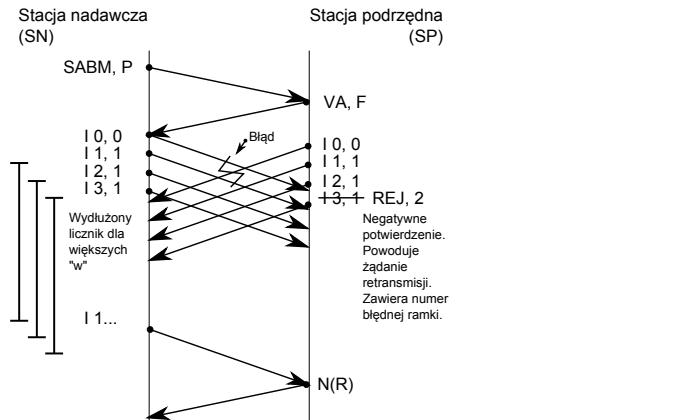
SN i SP mogą wysyłać jednocześnie. Brak sztywnego przypisania na stację podrzędną i nadzorowaną. RNR też tu działa.

#### 4.2.1 Mechanizmy potwierdzenia

- Pozytywne (stan NCR) - mówi, że nie trzeba się już zajmować którąś z ramek, która została potwierdzona.
- Negatywne (timeout) - wymusza retransmisję.

#### 4.2.2 Właściwości

- Parametr  $w$  pilnuje ilości przesyłanych ramek. Nadmiarowe muszą czekać i są przesyłane później.
- Sekwencja zakończenia podobna do poprzedniego trybu. Może być nawet RD, F, ale musi SP wysyłać potwierdzenia odebrania ramek od SN. Na końcu SN wysyła DISC.
- P i F są potrzebne tylko na początku.
- Timeout powinien być duży, ale nie za duży.



Wersja ze **SREJ** - retransmisja pojedynczej ramki



#### 4.2.3 Zrównoważony tryb działania (AMB (E) lub ABM - asynchronous balanced mode)

Dotyczy trybu asynchronicznego. E występuje dla dużych liczników.

To samo, tylko stacja, która zaczyna nadawać zostaje oznaczona jako "SN".

### 4.3 Inne

SDLC jest zastrzeżony przez IBM, dlatego powstały rozwiązania alternatywne (HDLC)

## 5 Jakość łącza

Czyli wpływ kontroli błędów na łącze.

### 5.1 Bit Error Rate (BER)

BER =  $10^{-4}$

Czasami zamiast tego liczy się FER - Frame Error Rate.

### 5.1.1 Przykład

Jest do przesłania 2 kB w blokach 256 B,  $w \equiv 3$

$$N = \frac{2[kB]}{256[B]} = 8 \quad (1)$$

Oznacza to, że:

- 1 na 10 tysięcy bitów zostaje przekłamanych
  - 1 na 1250 bajtów zostaje przekłamanych
  - 1 ramka 262 bajtów (6B nagłówka + 256B danych)
  - $\frac{1250}{262} = 3,45$  ramki
  - Co 3.45 ramki jest przekłamane.

**Rozwiązań**: więcej ramek przesyłanych (narzut organizacyjny protokołu). Wielkość ramek ↘→ Narzut ↘ W ↘→ Narzut ↘

## 5.2 Cechy

- Stopa błędów

$$q = \frac{\text{ilosc\_bledow}}{\text{ilosc\_bitow\_transmitowanych}} \quad (2)$$

- Pole danych w SDLC wynosi 128B, ale może też być 64, 256, 1024
- Licząc stopę błędy bierzemy całą ramkę wraz z flagami, więc do tego 128 dodajemy 6 (134)
- Optymalna długość pola danych

$$D_{opt} = \frac{H + C \times T_{out}}{2} \times \left[ \sqrt{1 - \frac{4}{(H + C \times T) \ln(1 - q)}} - 1 \right] \quad (3)$$

Gdzie:

- H - długość nagłówka
- C - przepustowość łącza

## 5.3 Przykład

Dane:

- $C = 9600[\text{bps}]$  (bit na sekundę)
- $T = 50[\text{ms}]$  (timeout)

Dla:

- $q = 10^{-3}$ ,  $D_{opt} = 715[b] = 89[B]$
- $q = 10^{-4}$ ,  $D_{opt} = 2262[b] = 282[B]$
- $q = 10^{-5}$ ,  $D_{opt} = 7155[b] = 854[B]$

**Wniosek:** im mniejsza stopa błędu tym dłuższe optymalne pole danych.

## 6 Łącze radiowe

- Dwa kanały: pierwszy z centrum komputerowego, drugi do CK.
- Problem dostępu do łącza: wbudowanie w terminal algorytmu dostępu do łącza:

### 6.1 Algorytm dostępu do łącza



Retransmisja następuje z losowym opóźnieniem by ramki ponownie się na siebie nie nałożyły.

## 6.2 Protokół

Zastosowano protokół swobodnego dostępu, ALOHA.

# 7 Komunikacja kablowa

## 7.1 Model

Jeden kabel, wiele stacji. Jak chce nadawać to nadaje, tylko czeka na potwierdzenie, jeśli nie nadaje, to dopiero retransmisja (algorytm swobodnego dostępu) (chyba).

## 7.2 Algorytmy

- CSMA (*Carrier Sense Multiple Access*) - wykrywa występowanie zakłóceń, objawiają się w postaci dwóch ramek. Po wykryciu końca kolizji decyduje, która ramka czeka, a która jest retransmitowana.
- p-CSMA (*persistent CSMA*)  
Punkt startu
- Algorytmy, które przerywają transmisję po wykryciu początku zakłócenia danej ramki
  - CSMA / CD - wykrywanie kolizji, początek ETHERNETu
  - CSMA / CA - zapobieganie kolizji, później, wcześniejsze zastosowanie w radiówce
  - CSMA / CR - rozstrzyganie kolizji, lepszy wygrywa i kontynuuje przesył, słabszy musi znów wysłać

# 8 Dostęp do łączna

## 8.1 Protokoły dostępu do łączna

- Swobodnego dostępu (np. ALOHA)
- Częściowo kontrolowanego dostępu (np. CSMA / ...)
- Kontrolowanego dostępu - faza wykonania bez kolizji i faza gdzie może być.  
Sposoby kontroli:
  - Planowanie transmisji
  - Przekazywanie żetonu (token)

## 9 Częściowo kontrolowany dostęp (działanie CSMA / ...)

### 9.1 Algorytm dostępu



Algorytm oparty o kilka prostych zasad

- "nie przeszukadzaj" (sprawdza czy jest widoczna nośna) - zaczynamy nadawać gdy kanał się zwolni CSMA persistent,  $P(\text{startu}) = 1$ . Możliwe kolizje.  
CSMA p-persistent,  $P(\text{startu}) = p$ . (jakoś tak)
- "czy koliduje?" (patrz wyżej)

### 9.2 Czas do retransmisji

Oznaczany jako  $T_R$

$$T_R = r(x) \times (2^k - 1) \times T_{ob}$$

Gdzie:

- $r(x)$  - liczba losowa z zakresu 0...1
- $T_{ob}$  - czas obiegu łącza, w najgorszym wypadku podwojony czas propagacji
- $k$  - liczba kolizji, czyli który raz się te ramki już zderzyły.
- Wraz ze zwiększaniem się liczby kolizji, możliwy zakres rośnie eksponentialnie. Jednak maksymalna wielkość  $k$  jest równa **10**. Dla  $k > 10$  przyjmuje się  $k = 10$ .
- Próbujemy szczęścia aż do  $k \leq 16$ , potem próby są przerywane.

### 9.3 Cechy

Maksymalnie 2800 m do pokonania.

Najgorszy czas  $47,2 \mu s$ .

$51,2 \mu s = 64B$  wyście. W tym czasie mogą się zdarzyć kolizje.

## 9.4 Poprawna transmisja

Ramka musi być dłuższa niż 64B. Taki wymóg ma Ethernet.

### 9.4.1 Opcje

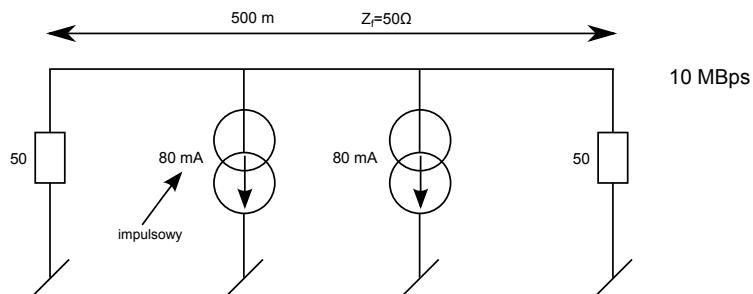
- 3 segmenty i 2 repeatory
- 3 segmenty i 4 pół-repeater

## Sieć typu Ethernet

### 1 Budowa

Standard Ethernet 2.0 wykorzystuje kabel koncentryczny.

#### 1.1 Kabel koncentryczny

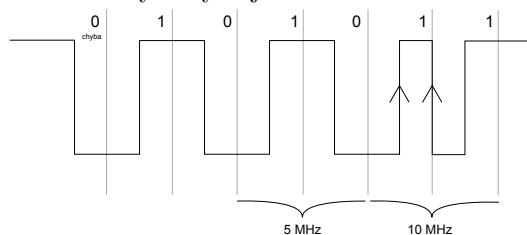


**Parametry:**

- $U_{IC} = U_p \times e^{\alpha l}$
- $\alpha l \leq 6dB|_{5MHz}$
- $\alpha l \leq 8.5dB|_{10MHz}$

#### 1.2 Kodowanie

Standard wykorzystuje kodowanie Manchester.



#### 1.3 Sprawdzanie kolizji

Chyba chodzi o zapobieganie kolizjom na poziomie sprzętowym.

- Sprawdzenie poziomu
- Pomiar wartości średniej prądu

- Wyłapywanie regularności szpilek
- $T_{ob} = 47.7\mu s$

W trakcie transmisji początkowych 64 bajtów ramki może być złe.  
Mogą wystąpić 3 segmenty kolizji +  $4 \frac{R}{2} + 2$  segmenty łączna.

## 2 Ramka

To jest (chyba) ramka dla CSMA/CD.

Element	Wielkość	Komentarz
Preambuła	8 B	$7 \times 10101010$ , a na końcu <b>10101011</b> .
Adres odbiorcy	6 B	MAC
Adres nadawcy	6 B	MAC
Kontrola	2 B	DIX Ethernet - typ protokołu; IEEE - długość danych
Pole danych	od 46 B do 1500 B	
CRC	4 B	
Cisza na łączu między znakami	12 B	$9.6 \mu s$

Każdy adres w Ethernecie powinien kończyć się zerem - adres stacji - jeden oznacza adres grupowy.

## 3 Zapobieganie kolizji

### 3.1 Historyczne metody

- Wprowadzenie Hubów - zmiana transmisji z szeregowej na równoległą.
- Switch: usuwa ramki kolizyjne. Zapamiętuje wysypane ramki, które może bez końca retransmitować  
→ Znosi odpowiednie rozpiętości sieci → wystarczy więcej switchów.
- Logiczny podział na małe sieci lokalne.
- Jumbo frame - pole danych do 9000 B. Wada: faworyzowanie pewnych stacji.

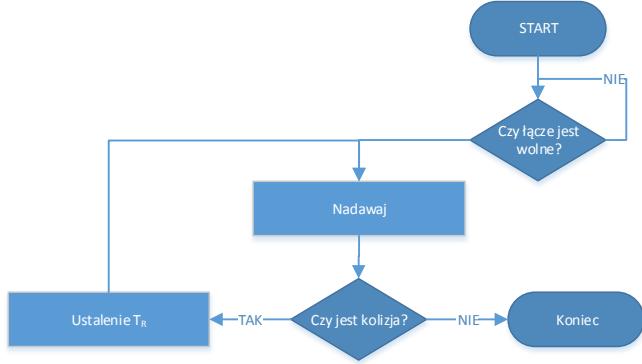
### 3.2 Klasyczny algorytm

#### 3.2.1 Idea

Kilku chce wysłać, czekaj, retransmitują, znów problem.

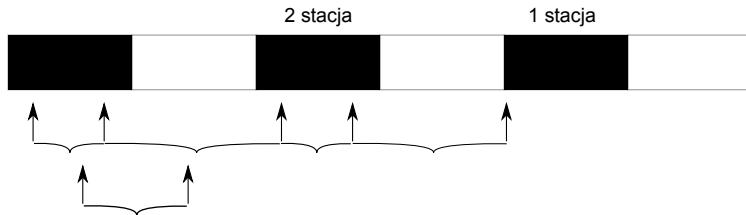
”Nie ma sensu czekać na wolne łączne, lepiej wygenerować pewien czas po którym ponownie sprawdzi się dostępność łączna”.

### 3.2.2 Schemat blokowy



### 3.3 CSMA/CA

Idea: unikanie kolizji



**Problem:** 1 stacja próbuje wysłać, 2 stacja próbuje wysłać.

**Analiza:** jak widać brak kolizji. Najpierw patrzy czy łącze jest wolne, jeśli nie to generuje czas losowy, do następnej retransmisji. Czyli na początku ta generacja czasu, a nie na końcu jak w klasycznym Etherencie.

#### 3.3.1 Ramka

	2B	2B	6B	6B	6B	2B	6B	0-23(?)B	4B
Preambula	Frame Control	Duration	A1	A2	A3	Nr sekwencji (?)	A4 (opcjonalnie)	Dane	CRC

#### 3.3.2 Problem

Duże zakłócenia przy dużej ramce.

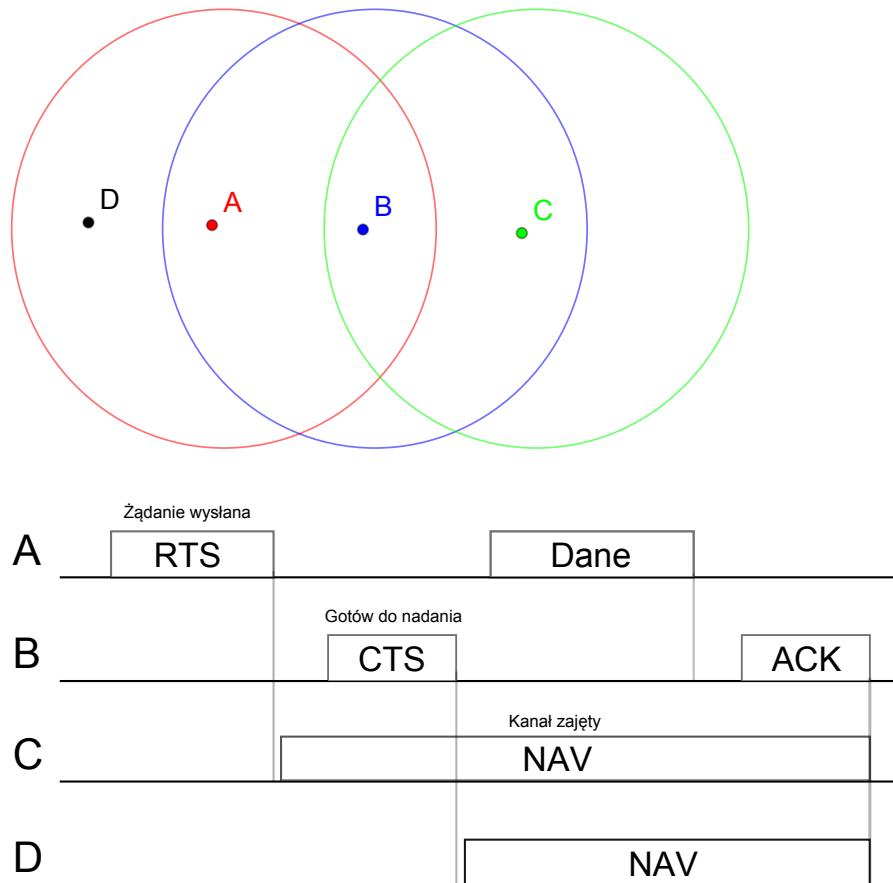
FC i NS wspomagają fragmentację ramki.

Wraca wzór Shannona z pestek:

$$C = B \times \log_2 \left( 1 + \frac{P_s}{P_w} \right)$$

### 3.3.3 Zmodyfikowane CSMA/CA

Wykorzystywane w radiówce.



Czyli blokujemy pozostałe stacje (C i D), by nie zakłócały.

Stację C blokuje stacja A, a stację D blokuje stację B.

Po stwierdzeniu, że kanał jest wolny, stacja musi poczekać jeszcze, bo może przyjść ACK.

Jeśli stacja ustawi w ramce MORE to NAV (channel busy flag) u innych stacji jest przedłużony o DURATION + ostatni ACK.

### 3.4 CSMA/CR

Umożliwia rozstrzyganie kolizji. Wykorzystywane w mikrosieciach (IIC ( $I^2C$ )) (HiFi).

Pierwsza część sygnału to pole adresu nadawcy. Najbardziej efektywny. Do sieci 1000 kB/s. S-7 m sieci.

Sygnał nadawcy musi się rozejść po całej linii i wrócić, co zabiera ogromną ilość czasu.

### 3.5 Protokoły MAC (*Media Access Control*)

- Swobodny dostęp (z potwierdzeniem)
- Algorytmy częściowo kontrolowanego dostępu CSMA (bez potwierdzenia)
- Algorytmy kontrolowanego dostępu (z potwierdzeniem / brak potwierdzenia dla transmisji danych)
  - Rezerwacyjne
  - Selekcyjne
  - Jawnego wskazania (tzw. Token passing)

### 3.6 Algorytmy kontrolowanego dostępu

- Algorytm z fazą planowania, gdzie jest dopuszczalne rozstrzyganie kolizji (z limitowaną kolizją).
- Algorytm z przekazywaniem uprawnienia, gdzie jest gwarantowany dostęp do łącza (przekazywanie żetonu (Token))

W tych algorytmach można wyróżnić dwie fazy:

- Planowania - widoczna dla wszystkich stacji, podzielona na szczeliny czasowe.
- Wykonania - superramka, uruchamia (...) szczelin czasowych

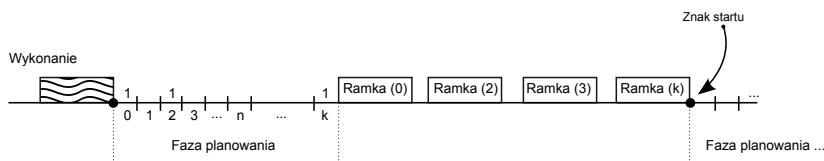
### 3.7 Algorytmy z fazą planowania

Dopuszczalne rozstrzyganie kolizji (z limitowaną kolizją).

#### 3.7.1 Algorytm rezerwacyjny

Rezerwacja prawa nadawania. Dzieli się na dwie fazy:

1. Rezerwację (faza planowania)
2. Wykonanie (faza wykonania)



**Faza planowania** - to są szczeliny czasowe.

- Każda stacja ma przypisanie do jednej z nich.
- Jeśli stacja chce transmitować to wstawia 1 w swojej szczelinie czasowej.
- Jeśli nie potrzebuje dostępu, to nie.
- Po 1 bit na każdą szczelinę czasową.

Zwykle grupuje się stacje i każda grupa otrzymuje po jednej szczelinie czasowej (wówczas liczba szczelin < liczba stacji).

Zawsze musi być minimum jedna ramka, bo musi być widoczny koniec transmisji.

**Działanie:** wysyłamy adres stacji z grupy, ale są kolizje. Tu włącza się metoda generacji czasu na retransmisję itp. Jeśli nie zdąży wysłać, bo już zacznie wysyłać, to czeka na ponowne pojawienie się szczelin czasowych.

W fazie planowania mogą wystąpić kolizje, ale dla każdej szczeliny kolizje są rozdzielane.

Wykorzystywany w sieciach o dynamicznych priorytetach.

#### 3.7.2 Algorytm selekcyjny

Selekcja wstępna, czyli która stacja ma prawo nadawania. Dzieli się na dwie fazy:

1. Selekcję (faza planowania) - tu rozstrzyga się, która ramka przypadnie na którą stację.
2. Wykonanie (faza wykonania)



Powyższy rysunek przedstawia malenie priorytetu.

Na jedną szczeleń może przypaść więcej niż jedna stacja (grupowanie).

WAŻNE: stacje, które muszą być wykonane są wysyłane jako pierwsze i do tego pojedynczo, a nie w grupie (jedna na jedną szczeleń). Mimo to, zwykle wykonuje się w grupach.

- Jeśli w trakcie nadawania superramki w sieci pojawi się nowa stacja, to musi ona poczekać do końca transmisji (początku kolejnej superramki), aby być uwzględniana w algorytmie.
- Problem pojawia się, gdy z sieci zniknie stacja w trakcie nadawania - nie wiadomo, kiedy znów nadaje daza planowania, a możliwe że na nieobecną maszynę przypadnie akurat jakaś ramka.
- Zazwyczaj stosuje się algorytmy hybrydowe, aby uniknąć takich sytuacji.

## 3.8 Algorytmy z przekazywaniem uprawnień (Token)

### 3.8.1 Co to jest Token?

Token to specjalna ramka sterująca.

Token bus jest zdefiniowany w IEEE.802.4.

### 3.8.2 Idea działania

Stacja przechowuje adres poprzednika i następcy. Powstaje wówczas lista adresów stacji:

$A_1 \rightarrow A_2 \rightarrow A_3 \rightarrow \dots \rightarrow A_n \rightarrow \dots \rightarrow A_k \rightarrow A_1 \dots$

Ilustracja przekazywania tokenu.

Parametry:

- Czas korzystania z tokenu:  $T_k$  40kbit
- No Token Timer:  $T_{NTokT} = n \times T_k$ , gdzie  $n$  to liczba stacji (?)

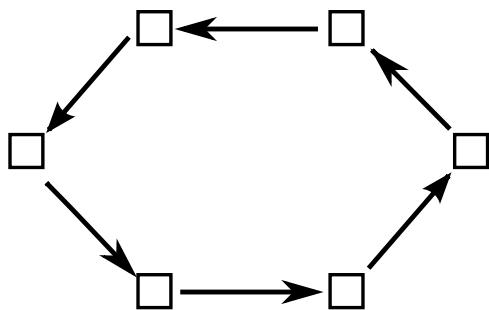
### 3.8.3 Wada

Wszystko się sypie gdy zostaje przeklamany Token. Stacja czeka, orientuje się, że sieć się sypła, a następnie stacje walczą o to, która stworzy nowy Token.

- Algorytm "głosowania Tokenu", w końcu któraś wygra.
- Algorytm włączania stacji do listy, czyli zgłaszanie następcy (ponowna walka).
- Problem opuszczania sieci.

## 3.9 Sieć pierścieniowa (Token Ring)

Zdefiniowana w IEEE 802.4 (?) Zastosowanie innej (pierścieniowej) topologii sieci. Eliminuje ona wcześniejsze problemy.

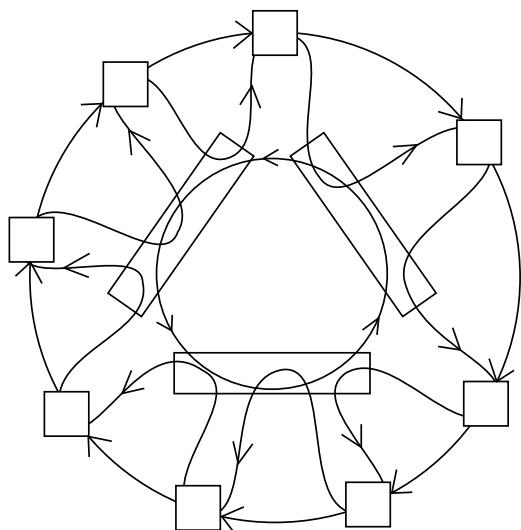


- Transmisja tylko w jednym kierunku. Dzięki temu sieć może mieć wiele kilometrów.
- Stacje nie wiedzą co się dzieje u innych.
- Ciężko w niej znaleźć miejsce przerwania.

### 3.9.1 Sieć pierścieniowa dla CSMA / CD

- Liczba bitów w pierścieniu  $N_{ring} = \frac{L}{\frac{2}{3}C} \times \frac{1}{V}$ , gdzie:  $L$  to długość kabla łącza;  $V$  to szybkość nadawania;  $C$  to prędkość światła
- Liczba bitów pamiętanych na kablach  $n = \frac{T_p}{T_b}$
- Dla Cambridge Ring:  $N_{ramek} = N_{ring} + N_{stacji} = 4N/40bit$

### 3.9.2 Sieć dwupierścieniowa (Token Ring)

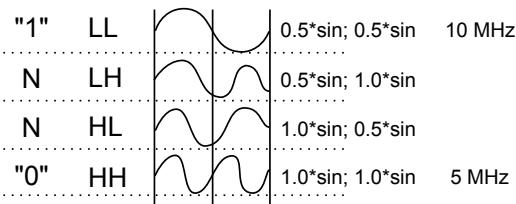


Dwa pierścienie: jeden zewnętrzny, drugi wewnętrzny (rezerwowy). Jeśli jeden przesyła w lewo, to drugi w prawo. Jeżeli zewnętrzny zostanie przerwany, to wewnętrzny zajmuje się załatwianiem tego.

**Wada:** jeżeli w sieci nie ma Tokenu to występują kolizje.

### 3.10 Kodowanie informacji

Kodowanie dla sieci opartych o Token.



### 3.11 Ramka

N	N	0	N	N	0	0	0	SD	start ramki
N	N	1	N	N	1	1	1	ED	koniec ramki

Te "N" mówią, że to SD lub ED, a nie zwykłe dane. Więc tu nie używamy metody szpikowania zerami i tym podobnych.

#### 3.11.1 Token Bus

SD	FC	DA	SA	Dane	CRC	ED
----	----	----	----	------	-----	----

Gdzie:

- SD - Start Delimiter, ma postać NN0NN000, 1B
- FC - Frame Control (czasami zwany też Flow Control), czy to transmisja w jednym kierunku czy np. żądanie przesyłu odpowiedzi. Przekazuje informacje sterujące. 1B
- DA - Destination Address (6B / 2B)
- SA - Source Address (6B / 2B)
- ED - End Delimiter, ma postać NN1NN111, 1B

#### 3.11.2 Token Ring

SD	FC	FC	DA	SA	Dane	CRC	ED	FS
----	----	----	----	----	------	-----	----	----

Gdzie:

- AC - adres rozpoznany, 1 - ok, 0 - nie (ramka skopiowana (jakoś tak)). Służy do przekazywania informacji sterujących jak np. wybranie pierwotnego posiadacza tokenu.
- FS - Frame Status

### 3.12 Budowa bajtu FC

#### 3.12.1 Budowa

Rodzaj ramki (2b)	Przeznaczenie ramki (3b)	PPP (3b)
-------------------	--------------------------	----------

### 3.12.2 Rodzaje ramki

- 00 - ramka kontrolna
- 01 - ramka z danymi
- 10 - ramka konfiguracyjna (station management, do ustalania budowy sieci)

### 3.12.3 Przeznaczenie ramki

- 000 - żądanie (command), brak odpowiedzi
- 001 - żądanie i odpowiedź
- 010 - odpowiedź

### 3.12.4 Bity PPP

- Dla ramki z danymi oznaczają one priorytet
- Dla ramek kontrolnych definiuje się tutaj rodzaj ramki.

## 3.13 Problemy związane z Tokenami (wadliwe stany)

- Brak Tokenu (najczęściej pojawia się przy samym starcie sieci)
- Więcej niż jeden Token
- Zabezpieczenie przy przekazywaniu tokenu (należy się upewnić, czy token został dobrze przekazany)
- Nagłe zniknięcie Tokenu - zdarza się, gdy stacja aktualnie przetrzymująca Token opuszcza sieć
- Nowa stacja w sieci (zmiany w liście stacji)

## 3.14 Timery (dostęp do łącza)

### 3.14.1 Parametry

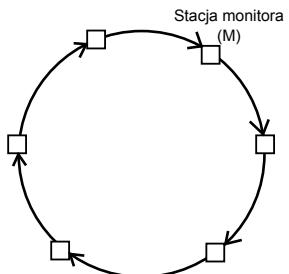
- **THT** (*Token Hold Timer*) - maksymalny czas przetrzymania Tokenu, równy  $n$  bitów
- **NTT** (*Not Token Timer*) - czas czekania na token.  $NTT = N \times THT$ , gdzie  $N$  to liczba stacji. Gdy trwa za długo to sieć pada.
- Problem pojawia się gdy przejdzie cały timer, a tokenu brak.
- Trzeba utworzyć listę adresów log (...). Każda stacja musi pamiętać dwa adresy: poprzednika i następnika.

## 3.15 Algorytm rozstrzygania kolizji

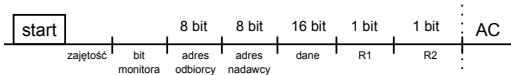
Dla Token Bus: stacja, która chce coś wysłać, w ramce w FC wysyła CT - zgłoszenie tokenu. Potem stacja walczy z innymi.

- W sieci pierścieniowej stacje robią opóźnienie 1 bit.
- Stacje walczą o dostęp wg algorytmu rywalizacji. Jeśli adres nadawcy jest niższy niż mój, to ja wysyłam własną. Jeśli jego wyższy to retransmisja (jego ramkę dalej ?).
- Po przejściu całej sieci ramka ma adres zwycięzcy. Przez to opóźnienia. Wtedy ramka dodatkowa, która mówi, że jest już zwycięzka i trzeba już normalnie przesyłać.
- W pierścieniowej technologii są potwierdzenia. Na końcu ramki dokleja się informację czy ramka była poprawnie wysłana i odebrana oraz czy zrobiono kopię tej ramki (ta odbierająca robi kopię). Na A lub C musi być jeden, że ok, lub zero, jeśli nie ok. Jeśli A i C = 1, to wszystko ok.
- Czyszczenie sieci pierścieniowej: nadawca coś wysyła, a jeśli coś do niej wchodzi w trakcie to są to śmieci, więc je usuwa. Usuwa też swoją ramkę po przejściu przez sieć.

### 3.16 Cambridge Ring

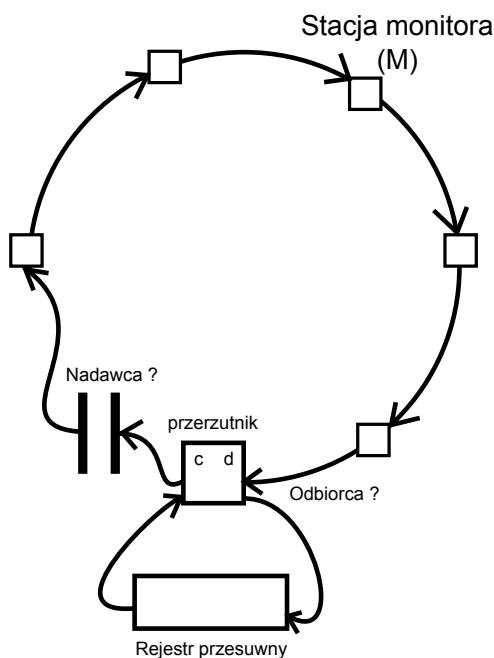


Pojemność informacyjna pierścienia  $20\mu s$  gdy  $4km$  sieci / 200 bit.  
Każda stacja M generuje miniramki o długości 40 bit.



- Sieć CR wykorzystuje monitor - wydzieloną stację.  
Monitor generuje "strumień" miniramek.  $Z = 1$ , czyli ktoś zajął tą ramkę i wpisał do niej dane (czyli używa się tej ramki do przesyłu).  
 $M = 1$ , czyli monitor już tą ramkę widział. Monitor skasuje tę ramkę gdy odczyta ją i ta ramka ma  $n = 1$ . Za drugim razem ??  
Gdy trzy pierwsze bity (startu, zajętości i monitora) są równe 1 to oznacza, że ramka obiegała cały pierścień i monitor wysyła nową tackę (??) informacyjną.
- Szybkość transmisji:  $\frac{2B}{T_{ob}}$ , gdzie  $T_{ob}$  to czas obiegu, który trwa  $30 \mu s$ .
- Najgorsze rozwiązanie pod kątem serwisowania.

#### 3.16.1 Algorytm włączania rejestru



Dzięki temu może przesyłać duże ramki. Część wchodzi do rejestru przesuwnego.

### **3.16.2 Protokół drzewa rozpinanego**

*Spanning Tree Protocol.* Przydatne przy sieci pierścieniowej z repeaterem, która posiada wiele podstacji. Jeśli występuje retransmisja to trwa ona w kółko dzięki zamkniętemu obiegowi.

## **Projektowanie topologii sieci**

Kolejnym problemem w komunikacji jest ograniczenie zasięgu sieci przez długość kabli lub zasięg transmisji. Potrzeba więc wykorzystać pośredników.

### **Przykładowe rozwiązania:**

- Proste: połączenie każdego z każdym, ale nie jest to efektywna metoda.
- Lepsze: każdy z elementów posiada minimum 2 połączenia - w razie awarii jednej ze stacji ma alternatywę.

W projektowaniu topologii związań jest kilka istotnych zagadnień.

## **1 Uzgodnienie gotowości udziału w transmisji**

### **1.1 Tryb połączeniowy**

#### **1.1.1 Idea rozwiązania**

Stacja A wysyła żądanie wyznaczenia trasy do stacji Z. Nazywamy to jako "pakiet żądania wykonania połączenia". Pakiet trafia do stacji B i jeśli ona może wysłać go dalej to odsyła potwierdzenie i szuka dalszej drogi. Fakt przesłania żądania zestawienia połączenia jest odnotowywany w każdym węźle przez który ten pakiet przechodzi (ID połączenia).

W końcu ten trafia do stacji Z i jeśli stacja Z się zgadza na połączenia to odsyła pakiet odpowiedzi pewną drogą (która wcale nie musi być zgodna z tą, którą przyszło żądanie) do stacji A. Dzięki temu, że w węzłach istnieją ID połączeń mamy wyznaczoną ścieżkę przez którą będą przebiegać komunikaty z A do Z, a nawet dwie.

#### **1.1.2 Pakiet**

ID połączenia	reszta
---------------	--------

#### **1.1.3 Cechy trybu**

W tym trybie nie ma problemu dublowania pakietów. Natomiast jeżeli jedno połączenie wysiądzie (np. na wskutek awarii stacji) to pojawia się poważny problem.

### **1.2 Tryb bezpołączeniowy**

#### **1.2.1 Idea rozwiązania**

Pakiet danych wysyłany wprost do sieci, niezależny przesył.

#### **1.2.2 Pakiet**

Adres odbiorcy	Adres nadawcy	reszta
----------------	---------------	--------

#### **1.2.3 Cechy trybu**

Tu mogą się zdarzyć duplikaty ramek itp. ale gdy któryś węzeł padnie to jest szansa, że wszystko dalej będzie działać.

W tym trybie działa większość sieci, w tym sam wielki Internet.

## 2 Tryb pracy sieci

### 2.1 Komutacja kanałów (*channel switching*)

Rezerwacja kanałów od Nadawcy do Odbiorcy. Na czas połączenia wszystkie połączenia od nadawcy do odbiorcy, włącznie z połączonymi pośrednimi, są zestawione i zablokowane. Jest związany z trybem połączeniowym.

### 2.2 Komutacja informacji (*information switching*)

Zamiast rezerwacji z góry wszystkich kanałów potrzebnych do komunikacji (komutacja kanałów) wyłącznie część między stacjami jest rezerwowana. Informacja jest przekazywana z węzła do węzła.

#### 2.2.1 Przykład

Wysyłamy ze stacji A do C przez B. Ze stacji A przesyłamy wiadomość do B i tylko ten fragment kanału jest zajęty. Stacja B potwierdza odbiór i teraz ona zajmuje się znalezieniem C - połączenie pomiędzy stacjami A i B jest wolne. Tu komunikat musi być zapamiętany przez stację pośrednią. Jest to komunikacja typu ***store & forward***

#### 2.2.2 Problem

Wielkość informacji - każda stacja musi ją zapamiętywać.

Rozwiązania:

- **Komutacja wiadomości** (*information switching*): zapis całości informacji w buforach w buforach itp. (chyba), limit 8kB, technologia ARPANET
- **Komutacja pakietów** (*packet switching*): dzielenie informacji na pakiety o stałym rozmiarze.

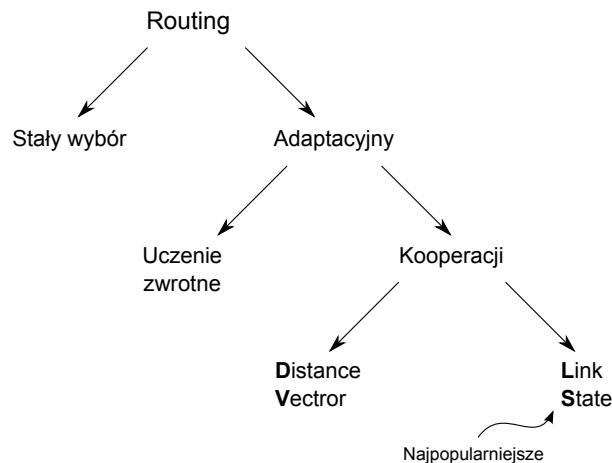
## 3 Adresacja

Wykorzystywana jest adresacja **IP**.

## 4 Wybór drogi - *routing*

### 4.1 Diagram rozwiązań

Najpierw był routing źródłowy (*source routing*), czyli ręczne wyznaczanie drogi przez nadawcę. Następnie pojawiły się rozwiązania w postaci algorytmów.



## 4.2 Nomenklatura

- Dużymi literami oznaczamy węzły
- Małymi literami oznaczamy ścieżki

## 4.3 Algorytmy stałego wyboru

Algorytm statyczny, drogę wyznacza administrator na sztywno.

## 4.4 Algorytmy adaptacyjne

Algorytm dynamiczny, każdy węzeł zbiera informacji potrzebne do wyznaczenia trasy. Decyzja jest podejmowana na podstawie informacji uzyskanych na podstawie przesłanych wcześniej pakietów. Każdy węzeł wyznacza z osobna gdzie iść dalej.

Z tym typem algorytmów związane są następujące zagadnienia:

### 4.4.1 Zdobywanie informacji

- **Uczenie zwrotne** - dodanie dodatkowego pola, gdzie przechowywane są informacje jak np. znacznik czasu (timestamp) wysłania pakietu. Dzięki temu węzeł wie jak długo szedł pakiet. następnie robi tabelkę. Np.

		Zi		
		B	C	K
Interfejs	et1	$t_{Be1}$		
	et2	$t_{Be2}$		
	et3	inf		

Na początku tablica jest pusta, co jest problemem. należy utworzyć początkową tablicę w jakiś sposób np. algorymem stałego wyboru. Jeżeli przyjąć, że każdy timestamp jest równy, to wtedy jako informację można przechowywać liczbę węzłów, przez którą się przechodzi (jednak prawie zawsze timestampy są różne).

Problem: konieczne są uaktualnienia tabelki. Jeżeli zabraknie update'u, wartość jest zatępywana przez nieskończoność (w praktyce: nie przesyłaj pakietów tą ścieżką). Początkowo tabela jest wypełniana samym nieskończonością, dlatego wraz ze startem należy użyć wartości wyliczonych przez inny algorytm (potem uczenie zwrotne staje się samowystarczalne).

- **Algorytm kooperacji (samodzielnego badania (?)**) - każdy węzeł odpytuje otoczenie, jakie węzły są obok niego, wysyła żądanie echo i otrzymuje odpowiedzi. Dzięki temu zna również czas. Echo musi czekać na obsługę jak każdy inny pakiet. Dzięki temu przesyłaniu (echo itd.) wychodzi metryka łącza.

Tutaj informacja ma swój czas życia ( $30 - 60 \mu s$ ). Jeżeli żaden pakiet nie przechodzi ta drogą to znaczy, że coś jest z nią nie tak.

Rodzaje:

- **distance vector** - wysyła pełną tablicę routingu do sąsiadów (drogi dostępne w otoczeniu i metryki)
- **link state** - przesyła do wszystkich węzłów informacje, że istnieje takie a takie łącze i metryka (koszt) tego łącza. Na podstawie tego, "wizualizowany" jest graf połączeń. W węźle sieci dostępny jest cały "obraz" sieci.

Zalety:

- \* Mniej problemów z pętlami w połączeniach (bo każdy węzeł ma obraz całej sieci)
- \* Łatwiejsza skalowalność

Wady:

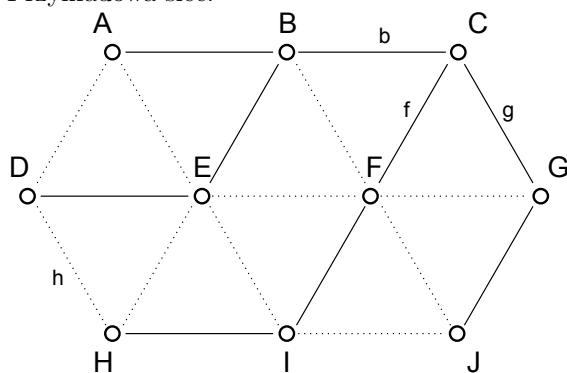
- \* Większe użycie CPU i pamięci w węźle
- \* Droższe i bardziej skomplikowane do implementacji i utrzymania

## 4.5 Inne algorytmy

- Zalewania (transmisji *broadcast*) - wysyłanie wszędzie w TTL. Bardzo nieoptymalny, ale zawsze działa (w końcu dotrze...). Pozwala ogarnąć nawet najbardziej zdezinformowaną sieć.
- Gorącego ziemniaka - wyślij do najbliższego / najlepszego z TTL (najmniej obciążonego) sąsiada. Problem: może stworzyć nieskończoną pętlę.

## 4.6 Wyznaczanie tablicy routingu

Przykładowa sieć.



Linie przerywane oznaczają brak drogi z jednej stacji do drugiej.

### 4.6.1 Graf drzewa spływu

Wyciąć te które mogą powodować zapętleń i potem szuka się tych dróg o najmniejszym koszcie.  
Przykład z tablicą decyzyjną dla stacji C:

Routing table

Adres docelowy	Decyzja	Koszt
A	b	2
B	b	1
C	-	-
D	b	3
E	b	2
F	f	1
G	g	1
H	f	3
I	f	2
J	g	2

Adres docelowy oraz decyzja tworzą wektor odległości (*distance vector*).

Kolumna decyzja to *routing*.

### 4.6.2 Algorytm stałego wyboru

Wykorzystuje *tablicę stałego wyboru*. Faworyzuje pewne drogi przed innymi.  
 $r(x)$  - generator liczb losowych, który mówi, którą trasę wybrać. jest to próba rozproszenia ruchu po sieci.

Routing table

	Decyzja 1	Decyzja 2	Decyzja 3
	$x \leq 0.7$	$0.7 < x \leq 0.9$	$x > 0.9$
A	b	f	g
B	b	f	g
C	-	-	-
D	a	e	c
E	a	e	c
F	f	g	b
G	f	g	b
H	f	b	g
I	f	b	g
J	f	b	g

Tabela posiada kolumnę z jedną główną trasą, która ma największe szanse się pojawić. Pozostałe dwie, mniej optymalne, są rezerwowe i mają mniejsze prawdopodobieństwo. Droga jest stała i ustalona z góry (chyba).

#### 4.6.3 Algorytm zwrotnego uczenia

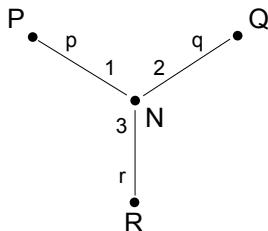


- $t_{Aa}$  - przekazanie pakietu do D z A przez a.
- Algorytm nie potrafi wystartować od zera, dlatego na niemożliwe ścieżki wstawia się nieskończoność.
- Informacja jest stale aktualizowana. Mówiąc o ile czasu potrzeba by wysłać pakiet różnymi ścieżkami.
- Przy przesyłce w drugą stronę wybiera się trasę najszysząszą. Nie musi być to pierwotna → jeśli istnieje inna szybsza, tedy idzie.
- Węzeł chce nauczyć się topologii sieci. Zapisuje z każdego przychodzącego pakietu informację o czasie przesyłu. Jeśli nic nie przychodzi to oznacza ścieżkę jako niedostępną.
- Węzeł nie posiada szczegółowych informacji o ścieżce, jedynie do którego sąsiada ona prowadzi.
- Udowodniono, że zestaw dróg lokalnie optymalnych powinien pokrywać się z trasą globalnie optymalną.

#### 4.6.4 Algorytm kooperacyjny z *distance vector*

Znany jako algorytm Bellmana-Forda.

- Wymiana metryk tylko z najbliższymi sąsiadami.



Gdzie:

- Liczby to metryki
- Duże litery to adresy
- Małe litery to ścieżki

Można z tego ułożyć ładną początkową tablicę.

- Następnie wezel  $N$  otrzymuje tablice od sąsiadów oraz dodaje sobie koszt dotarcia do sąsiadów.

P		Q		R	
Adres	Metryka	Adres	Metryka	Adres	Metryka
A	3	B	3	A	4
C	5	D	5	B	7
D	7	R	3	C	3
R	4	S	5	D	2
S	7				
Q	8				

+1                                    +2                                    +3

- W efekcie czego stacja  $N$  otrzymuje tablicę możliwości połączeń z innymi, dalszymi stacjami oraz wylicza sobie najkrótsze ścieżki (albo "interfejsy").

N		
Adres	Metryka	Ścieżka
A	4	p
B	5	q
C	6	p
D	5	r
P	1	p
Q	2	q
R	3	r
S	7	q

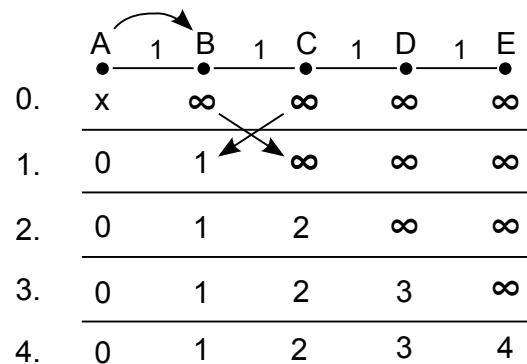
- Wymiana informacji w stałych odstępach czasu, 30 - 60 s, nie za szybko, nie za wolno.
- Gdy koszt jest taki sam na dwóch i więcej trasach to teoretycznie można wybrać dowolną z nich.
- Adres i metrykę nazywamy wektorem odległości, który jest wysyłany do sąsiadów.

## 5 Uszkodzenia w sieci

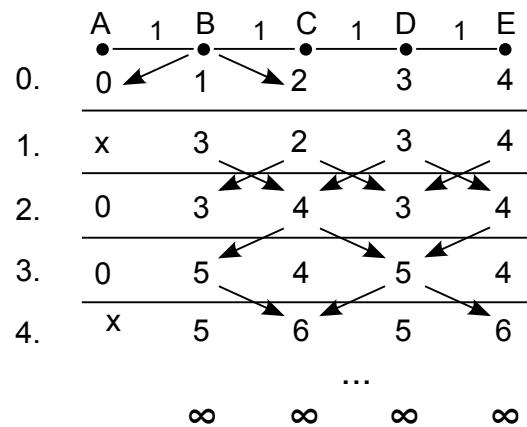
Wszystkie powyższe algorytmy działają w założeniu, że sieć jest sprawna oraz interesujące nas stacje działają i nie są uszkodzone. Jednak pojawia się kolejny problem: jeżeli któraś ze stacji zostanie uszkodzona i nie może przesyłać wiadomości, inne stacje muszą zostać o tym powiadomione. Analogicznie powinny być informowane o włączaniu stacji do sieci.

## 5.1 Rozchodzenie się informacji

### 5.1.1 Włączenie stacji



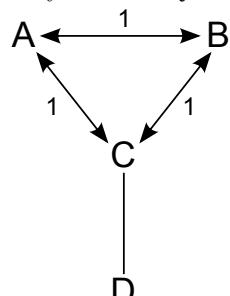
### 5.1.2 Odłączenie stacji



Powoduje liczenie do nieskończoności.

## 5.2 Rozwiążanie - Link State

- Zasada podziału horyzontów - określanie kierunków
- Nie są przesyłane niepotrzebne węzły (?)
- Przykładowe łącze



### 5.2.1 Stan łącza

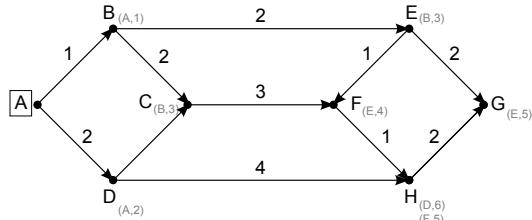
Nadawca	W
Nr sekwencyjny	
Wiek	
A	1
B	3

- Nr sekwencyjny - jeśli ktoś dostanie pakiet o większym numerze, to go przetwarza, a jeśli nie jest większy, to nie.
- Wiek - czas życia pakietu (w cyklu). Jeśli czas życia upływa to węzeł usuwa węzeł W ze swojej tablicy routingu. Liczony w sekundach (chyba)

### 5.2.2 Etapy Link State

1. "Zapoznaj się z sąsiadami" - wysłanie pakietu HELLO, który zawiera identyfikację nadawcy, oraz czekanie na takie same.
  2. "Wyznacz metrykę trasy" - Echo Request & Echo Response
  3. Algorytm dystrybucji pakietów LS
- Tworzenie tablicy decyzyjnej routingu (algorytm Dijkstry)

Przykładowa sieć, dane i kierunki dla stacji A.



Jeżeli mamy 2 różne wartości, np.  $A \rightarrow D - 3$ ,  $A \leftarrow D - 2$ , to przypisujemy średnią.

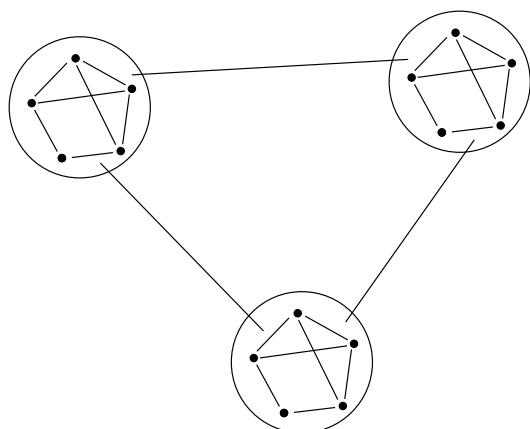
- Tablica routingu

Adres	Droga
B	b
C	b
D	d
E	b
F	b
G	b

- Zmiany w topologii są rozsypane natychmiast - nie ma liczenia do nieskończoności.
- Algorytm co jakiś czas wykonuje się ponownie (odświeżanie).

## 6 Problemy z liczbą węzłów

### 6.1 Metoda hierarchii



Jest tylko 1 wpis na całą sieć w tabeli routingu.  
Zawiera wpisy na połączenia lokalne.

## 6.2 Inny sposób



System autonomiczny. AS1 - AS3 (mogą to być firmy, instytucje itp.) traktujemy jako osobne węzły, więc rysunek można uprościć do postaci zwykłego trójkąta.

## 6.3 Routery

Dzielimy je na:

- Zewnętrzne - EGP (External Gateway Protocol)
- Wewnętrzne - IGP (Internal ...)

IGP działa dla routingu w najbliższym sąsiedztwie, natomiast EGP ustala zasady routingu między sąsiedztwami.

## 6.4 Przeciążenie Sieci

### 6.4.1 Oznaki przeciążenia sieci

Wcześniej muszą zostać zdefiniowane (którego używamy).

- Przeciętna długość przerwy pomiędzy transmisją kolejnych pakietów.
- Retransmisja pakietu ze względu na timeout
- Przeciętna długość que[ry] (????)

### 6.4.2 Poziomy zagrożenia

- Dzielą się na: zielony, żółty, pomarańczowy, czerwony. Osiągnięcie pomarańczowego poziomu za-wyczaj oznacza problemy w sieci.
- Inna notyfikacja: Overload Forward Congestion Notification (OFCN) - obliczana procentowo. Kolejne poziomy obciążenia obliczane są od górnej granicy (100%, 50%, 25%...)

### 6.4.3 Ogólna zasada działania

Generowana jest pula wolnych biletów. W sieci może krążyć tylko tyle pakietów, ile jest wolnych biletów. Następne pakiety muszą czekać, aż zwolni się jakiś bilet. Po upływie pewnego czasu generowana jest nowa pula biletów.

#### 6.4.4 Algorytmy zapobiegające obciążeniu

- Random Error Discard/Detection (RED) - wyrzuca się losowe pakiety, żeby zmniejszyć tłok na łączu.
- Error Recovery (algorytm retransmisji) - Aktywowany, kiedy pakiety przestają się mieścić w czasie timeoutu. Tworzy się retransmisję lub retransmisję selektywną (konkretnego pakietu) - jeżeli ponownie nie trafi z pakietem, to jeszcze bardziej obciąży się.

## Protokoły

### 0.5 IPv4

#### 0.5.1 Klasy IP

- Nazwa Klasy - zapis / maska bitów
- A = 10.000 / 8
- 16-B 172.16 - 31.255.255 / 12
- B 192.168.0.0 / 16

#### 0.5.2 Transport

IP	Port (16)
----	-----------

APIPA *Automatic Private IP Assigned* - algorytm uruchamiany, kiedy system nie może pobrać adresu z sieci, a ma pobrać adres z sieci. 169.254.0.0 - generowany jest losowy adres, który przepisywany jest systemowi. Komputery, które są odłączone od sieci automatycznie konfigurują się tym algorytmem. Kiedyś komputery nie mogły się uruchamiać bez adresu.

Broadcast, przykład:

27(maska)	5
-----------	---

- 00000 - Network
- 11111 - Broadcast

### 0.6 IPv6

Rozmiar 128 bitów, znacznie więcej niż w IPv4.

ver	PRID	Etykieta przepływową
Długość danych	Next Header	HopLimit
SourceAddress (128 bit)		
Destination Address		

1: 0, 2, 24, 31

2: 8, 8, 8

Next headers:

- Hop by hop options - opcje wykonywane w kolejnych krokach
- Routing

- Fragmentation
- Authentication
- Destination options
- Encryption security payload
- Jumbo header - nagłówek sygnalizujący specjalny rozmiar pakietu, 1 MB.

### **Routing**

Next Header określa rodzaj.

Next header	Pole związane z nagłówkiem	Kod opcji (ilość adresów + network adres)
x	BitMap	
1 do 24 Adres IPv6		

### **Jumbo**

Dla środowiska superkomputerów RFC 1883 - 1887.

Next header	Pole związane z nagłówkiem	194	
Jumbo payload size			

- Unicast address - do jednego
- Multicast address - do wielu
- Anycast address - do jednego z wielu (w obrębie grupy)

2001:cdba:0000:0000:0000:3257:9652 - pełny 128-bitowy adres IP.

CDBA - heksadecymalnie, określają 4 grupy.

W zapisie grupa 4 zer może być zapisana jednym zerem.

2001:cdba:0:0:0:3257:9652

2001:cdba::3257:9652

#### **0.6.1 Specjalne adresy IPv6**

- **::/96** - adres zgodny z zapisem IPv4
- **::/128** - odpowiednik adresu IP składającego się z samych zer i oznacza sieć nieokreśloną (*Unspecified Address*).
- **::1/128** - adresem local loopback jest adres składający się z samych zer i jednej jedynki.
- **2001:db8::/32** - *documentation prefix*
- **fec0::/10** - *side local prefix* - dla pierwszych 16 bitów, jeżeli są one w tym, to definiuje adresy używane wewnątrz danej lokalizacji.
- **fc00::/7** *unic local address*
- **ff00::/8**
- **ff80::/10** - adres przydzielony w łączu (*link local prefix*)

## 0.7 Protokoły

### 0.7.1 TCP

Budowa headera:

(Preheader)Source IP			
(Preheader)Destination IP			
Source port	Destination port		
Nr sekwencyjny pakietu			
nr potiwerdzenia pakietu			
TCP header length	x	Flagi	Window Size
Suma kontrolna		Urgent pointer	
OPCJA			
SAC			

Flagi TCP:

- FIN
- SYN
- RTS
- PUSH
- ACK
- URG
- ECW
- CWR

### 0.7.2 UDP

Source port	Destination port
Packet length	Suma kontrolna

Flagi i komendy UDP:

- URG
- ACK
- PSH
- RST
- SYN
- FIN

**Algorytmy, które wymuszają pewne mechanizmy zapobiegające przeciążeniu sieci.** odpowiadają za zabezpieczenie transportu z warstwy trzeciej do warstwy czwartej - gubieniu pakietów.

- slow start
- congestion avoidance
- multiplicative decrease

Obrazek: slow start Uzgodniony rozmiar okna (z odbiorcą), którego nie możemy przekroczyć.

### 0.7.3 SMTP (Simple Mail Transfer Protocol)

**Przykład sesji:**

1. Nawiązanie połączenia z serwerem. Polecenie "helo"
2. Podanie adresu nadawcy. Polecenie "mail from ...."
3. Podanie adresu odbiorcy. Polecenie "mail to ...."
4. Rozpoczęcie podawania wiadomości. Polecenie "data"
5. Podanie treści wiadomości.
6. Zakończenie treści znakiem "." w nowej linii.
7. Rozłączenie z serwerem. Polecenie "quit"

W oryginalnym protokole, dane wysyłało się do lokalnego MTA (Mail Transfer Agent), który sam już przesyłał dane przez łańcuch serwerów, aż do docelowego MTA. **Problem** z SMTP był taki, że nie istniał mechanizm autoryzacji nadawcy, wprowadzono poprawkę w postaci protokołu SMTP-AUTH, który wymagał autoryzacji nadawcy na jego MTA, ale nie rozwiązywało to problemu do końca.

### 0.7.4 POP (Post Office Protocol)

Dla zwiększenia bezpieczeństwa transmisji (i kontroli otrzymywanych wiadomości) opracowano protokół POP.

#### 0.7.5 Parametry czasowe

- refresh - czas odświeżania, co jaki czas secondary pyta primary
- retry - czas po którym następuje kolejna próba połączenia, powtórzenie refreshu, aż się pliki pobierze, albo aż minie expire.
- expire - czas po którym połączenie się oddaje
- TTL - czas życia w pamięci notatnikowej, jak DNS go pobierze do na tyle czasu trwa w DNS

#### 0.7.6 Name Serwery

- Primary
- Secondary
- Resolver - klient DNS
- Couching - primary lub secondary. Serwer notatnikowy.

## 1 Poczta elektroniczna

### 1.1 Historia

Rok 1982: rdc 822 - definiuje metody komunikacji między klientami, tzw. systemy agentowe. Powstał jako wersja przejściowa wielkich standardów ISO.

- user-agent
- mail-transfer-agent

Bardzo trwała prowizorka.

## 1.2 STMP

*Simple Transfer Mail Protocol.* Przypisany do portu 25.

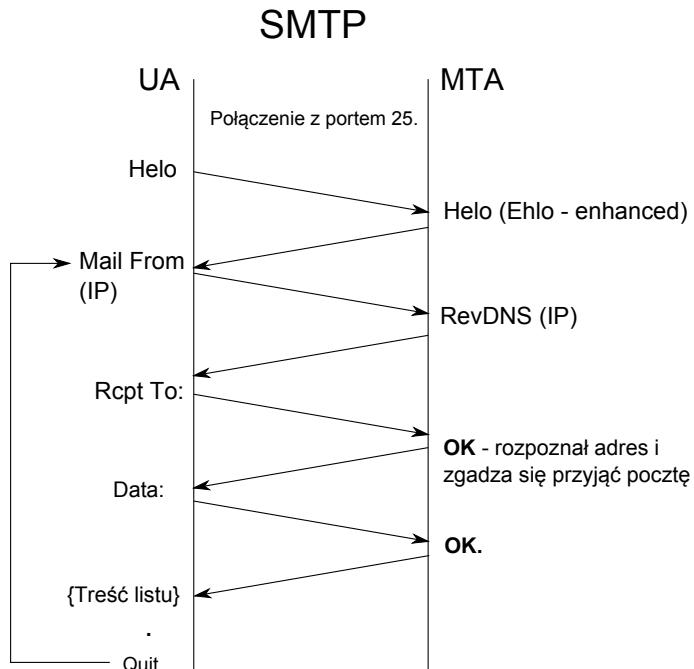
### 1.2.1 Adres użytkownika

- Zbiór par: atrybut = wartość

### 1.2.2 przykład

|  $C = US$  |  $ST = NewYork$  |  $L = WestPoint$  |  $PA = 360\ Memorialdrive$  |  $CN = TomSmith$  |  
Gdzie:

- SL - slot
- L - locality
- PA - present address



### 1.2.3 Nagłówki

Nagłówek: [do 1024 znaków][znak ENTER]

- To:
- Cc: (*carbon copy*) - wysyła kopie listu
- Bcc: (*black carbon copy*) - to co wyżej, ale odbiorcy są niewidoczni (??)
- From:
- Sender: - autor listu np. sekretarz, nie prezes
- Reply-to:
- Received:
- Return-path: - informacja pozwalająca odesłać przesyłkę

#### 1.2.4 Indentyfikatory

- Message Id: unikalne id przesyłki nadane przez serwer
- In-Reply-to: identyfikator przesyłki do której przesyłający się odwołuje
- References: id jeszcze wcześniejszych przesyłek
- Keyword: słowa kluczowe dla danej przesyłki, sprawy
- Subject: temat, 1 linijka, podsumowanie przesyłki
- X- : dodatkowy tekst na możliwe nowe nagłówki dla konkretnych aplikacji

Cały standard jest oparty o standard ASCII. Wraz z rozwojem sieci i internetu pojawiła się potrzeba rozwoju poczty.

rfd 1341, potem 2045-2049.

Multipurpose Internet Mail Extension (MIME).

- 5 dodatkowych nagłówków
  - **MIME-version:** - oznaczenie standardu MIME
  - **Content-description:** - słowy opis co to jest za zawartość
  - **Content-Id:** - identyfikator zawartości (?)
  - **Content-Transfer-Encoding:** - sposób zakodowania przesyłu
  - **Content-Type:** - informacja co to za zawartość i jak ją interpretować
- Kilka dodatkowych algorytmów związanych z transferem

#### 1.2.5 Kodowanie transferu

- 7-bit ASCII
- 8-bit ASCII
- BINARY
- Dwa ostatnie niosą ze sobą ryzyko związane np. z polskimi znakami.
  - BASE-64
    - \* a-z
    - \* A-Z
    - \* 0-9
    - \* + /
    - \* razem 64 znaki.
  - Quoted Printable Encoding - drukowalne przesłanie.
    - \* Jeżeli w treści listu jest mniej znaków narodowych to przepychamy je inaczej, za pomocą dwóch znaków np.
    - \* np. ą = 9Dh = 39 45 = "9", "d"
    - \* Jest to algorytm przekodowywania treści

#### 1.2.6 Typy zawartości

Typ	Podtyp
Text	Plain, Enriched, HTML, XML
Image	gif, jpeg
Audio	basic, mpeg (rfe 3003)
Video	mpeg
Application	octet-stream, postscript, pdf
Message	rfe822, partial, external-body
Multipart	mixed, alternative, parallel (np. równoczesne odczytywanie video i audio), digest

## 1.3 Rozszerzenia MIME

### 1.3.1 S-MIME

Security MIME, stworzone do przekazywania zakodowanych zawartości.

Typy zawartości:

Typ	Podtyp
multipart	signed (podpisany)
application	(dla danych kryptograficznych) pkcs7-mime - signed Data; pkcs7-mime enveloped Data; pkcs7-degenerated signed Data - PKI certyfikat; pkcs7-signature - podpis elektroniczny; pkcs10-mime - żądanie wystawienia certyfikatu (wstawka, która pozwala przekazać dane do systemu, który generuje klucze i tworzy certyfikaty)

## 1.4 HTML

Zdefiniowany tak jak protokół HTTP. Przydzielono mu nr portu 80. Case-sensitive - rozróżnia się duże i małe litery.

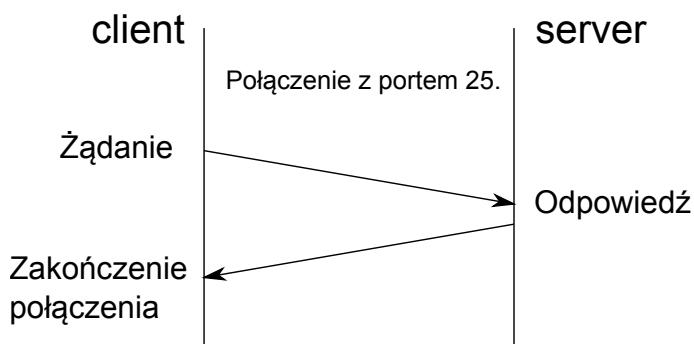
### 1.4.1 Metody

- GET - nazwa pliku + wersja protokołu (np. HTTP/1.0)
- HEAD - pozwala na ściągnięcie dokumentu
- POST
- PUT - załadowanie na serwer dokumentów (zdalna aktualizacja)
- DELETE - usuwania wskazanych dokumentów
- OPTIONS
- TRACE - jak ping, sprawdza czy dokument wysłany do serwera po powrocie nie podlega modyfikacjom
- CONNECT - brak zastosowania XDDDDD
- PATCH

### 1.4.2 Połączenie

Standard zakłada pojedyncze połączenie i rozłączenie się.

## HTML



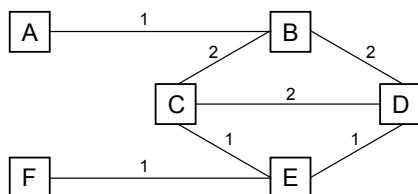
Zawiera się tu całość zawarta w `<body> ... </body>`. Dokument zawiera szereg nagłówków (*message headers*), które zawierają informacje identyfikujące klienta przed serwerem oraz serwer przed klientem. Udziela informacji jakie wersje MIME jest w stanie obsługiwać, jakiego rozmiaru dane klient może pobierać (jakimi porcjami), mówi też kiedy dokument był modyfikowany.

# Zadania

## 1 Topologia sieci

### 1.1 Zadanie

#### 1.1.1 Treść



W sieci z metryką jak na rysunku routery wymieniają się informacjami co 40 sekund, stosowany jest protokół typu Distance Vector. W chwili  $t_0$  router F ulega awarii. Określ, po jakim czasie informacja o niedostępności routera F dotrze do węzła A.

#### 1.1.2 Odpowiedź

	A	B	C	D	E	F	
0	5	4	2	2	1	0	Najpierw tworzymy tabelę z kosztami dotarcia informacji z poszczególnych węzłów do F.
1	5	4	2	2	3	X	Następuje awaria F. Węzeł E nie może się dostać do niego, więc próbuje przez C lub D, bo wie, że mają do F koszt równy 2. Więc $1 + 2 = 3$ .
2	5	4	4	4	3	X	E się zaktualizował, jednak drogi C i D do F zależały od E, dlatego one też muszą się zaktualizować. Ponieważ obecnie E do F ma 3, one też tyle dodają, $1 + 3 = 4$ .
3	5	6	4	4	5	X	Teraz B widzi, że C i D się zaktualizowały, a od nich zależała droga z B do F, więc ten też robi update. $2 + 4 = 6$ . Równocześnie E widzi tę aktualizację, a od C i D zależy jego droga do F (po zerwanym połączeniu), więc dodaje nowy koszt: $1 + 4 = 5$ .
4	7	6	6	6	5	X	Teraz wreszcie A widzi aktualizację B, od którego zależy jego droga do F. Uaktualnia się, $1 + 6 = 7$ . To samo robią C i D, których droga do F zależy, po awarii, przez B.
5	7	8	6	6	7	X	I w tym momencie wpadamy w pętlę aktualizacji.
6	9	8	8	8	7	X	
7	9	10	8	8	9	X	
8	11	10	10	10	9	X	
9	11	12	10	10	11	X	
10	13	12	12	12	11	X	
11	13	14	12	12	13	X	
12	inf	14	14	14	13	X	Tutaj następuje koniec obliczeń. W standardzie $\geq 15$ oznacza nieistniejący węzeł i oznaczany jest jako nieskończoność.

A więc stacja A dowie się o niedostępności stacji F po  $12 \cdot 40 \text{ s} = 480 \text{ s}$ .

## 1.2 Zadanie

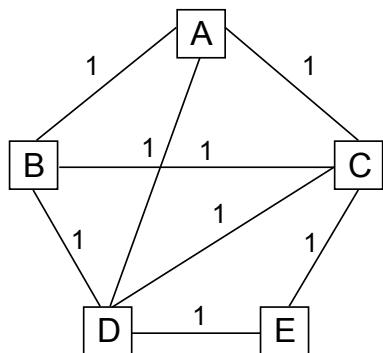
### 1.2.1 Treść

Dana jest następująca topologia sieci:

- Węzeł B ma jako sąsiadów węzły A, C i D
- Węzeł C ma jako sąsiadów węzły A, B, D i E
- Węzeł D ma jako sąsiadów węzły A, B, C i E
- Węzeł E ma jako sąsiadów węzły C i D

W chwili  $t_0$  węzeł B ulega awarii. Przedstaw sposób rozchodzenia się tej informacji między węzłami, po jakim czasie informacja o tym dotrze do węzła E, jeśli w sieci stosowany jest protokół RIP typu Distance Vector i cykl wymiany wektorów wynosi 30 s. Przyjmij metrykę jednostkową łączy.

### 1.2.2 Odpowiedź



Zmiana	A	B	C	D	E	Komentarz
0	1	X	1	1	2	Moment awarii
1	2	X	2	2	2	B ulega awarii. Sąsiedzi, czyli A, C i D próbują połączyć się z swoimi sąsiadami.
2	3	X	3	3	3	Węzły aktualizują się wzajemnie oraz E musi zrobić swoją.
3	4	X	4	4	4	I zaczyna się pętla.
4	5	X	5	5	5	
5	6	X	6	6	6	
6	7	X	7	7	7	
7	8	X	8	8	8	
8	9	X	9	9	9	
9	10	X	10	10	10	
10	11	X	11	11	11	
11	12	X	12	12	12	
12	13	X	13	13	13	
13	14	X	14	14	14	
14	15	X	15	15	15	Koniec.

Obliczenia kończą się, gdy koszt trasy z węzła E do B przekroczy 15. Wówczas widzimy, że dokonanych zmian było 14, a więc czas wynosi:  $14 \cdot 30 \text{ s} = 420 \text{ s} = 7 \text{ minut}$ .

## 2 Ethernet

### 2.1 Zadanie

#### 2.1.1 Treść

Kontroler karty sterownika sieci Ethernet odebrał następujące ramki:

1. Długość 51 bajtów, poprawne CRC
2. Długość 39 bajtów, błędne CRC
3. Długość 66 bajtów, poprawne CRC
4. Długość 1510 bajtów, poprawne CRC
5. Długość 1525 bajtów, poprawne CRC
6. Długość 1540 bajtów, błędne CRC

Jak zaklasyfikować powyższe ramki? (błąd protokołu, błąd transmisji, kolizja, poprawna ramka)

#### 2.1.2 Odpowiedź

- Jeśli ramka ma mniej niż 64 bajty i błędne CRC to jest to kolizja
  - Jeśli ramka ma mniej niż 64 bajty i poprawne CRC to jest to błąd protokołu
  - Jeśli ramka ma więcej niż 1518 bajtów, to zawsze jest to błąd protokołu, niezależnie od poprawności CRC
  - Jeśli ramka ma niemniej 64 bajty i nie więcej niż 1518 bajtów oraz niepoprawne CRC to jest to błąd transmisji
  - Poprawna ramka powinien mieć 64 - 1518 bajtów i poprawne CRC
1. Błąd protokołu
  2. Kolizja
  3. Poprawna
  4. Poprawna
  5. Błąd protokołu
  6. Błąd protokołu

## 2.2 Zadanie

### 2.2.1 Treść

W przypadku wystąpienia kolizji w segmencie sieci Ethernet czas oczekiwania na retransmisję  $T_R$  generowany jest jako liczba losowa z przedziału:

1. O stałej wielkości ustalonej przy konfiguracji sieci (jakiej?)
2. O wielkości rosnącej liniowo z numerem kolizji
3. O wielkości rosnącej z kwadratem numeru kolizji
4. O wielkości rosnącej z kwadratem do pewnego numeru kolizji

Jak obliczany jest czas  $T_R$ ?

### 2.2.2 Odpowiedź

Chodzi tutaj o algorytm CSMA, wykorzystywany do rozwiązywania kolizji na łączu kablowym. Wzór, o który prosi pytanie, zawarty jest w algorytmie dostępu w transmisji częściowo kontrolowanej. W pełni kontrolowanym dostępie występuje Token oraz rywalizacja stacji o dostęp (chyba o to chodzi).

$$T_R = r(x) \times (2^k - 1) \times T_{ob}$$

Gdzie:

- $r(x)$  - liczba losowa z zakresu 0...1
- $T_{ob}$  - czas obiegu łącza, w najgorszym wypadku podwojony czas propagacji
- $k$  - liczba kolizji, czyli który raz się te ramki już zderzyły.
- Wraz ze zwiększaniem się liczby kolizji, możliwy zakres rośnie eksponentialnie. Jednak maksymalna wielkość  $k$  jest równa **10**. Dla  $k > 10$  przyjmuje się  $k = 10$ .
- Próbujemy szczęścia aż  $k < 15$ , potem próby są przerywane.

Dlatego poprawna odpowiedź to: *Czas oczekiwania na retransmisję  $T_R$  generowany jest jako liczba losowa z przedziału o wielkości rosnącej z kwadratem do pewnego numeru kolizji, a tym numerem jest 10*  
Forczo: jeszcze to sprawdzę.

## 2.3 Zadanie

### 2.3.1 Treść

Chcemy zastosować protokół CSMA / CD przy budowie sieci o długości łączka 500 m przy szybkości transmisji 200 MBit/s. Jaka powinna być w tym przypadku minimalna długość ramki?

### 2.3.2 Odpowiedź

Najpierw liczymy czas propagacji

$$t_p = \frac{dystans}{predkosc\_lacza}$$

Predkości łączka nie mylić z szybkością transmisji. W przypadku światłowodu wynosi ono 2/3 predkości światła czyli 200 000 000 m/s, tu takie założenie można dać. O tej wartości należy pamiętać na sztywno. Następnie liczymy czas propagacji sygnału przez 500 [m]:

$$t_p = \frac{500 \text{ [m]}}{200 000 000 \text{ [m/s]}} = 2.5 \mu s$$

W obie strony czas ten wynosi  $2.5 \mu s = 5 \mu s$ .

Bierzemy czas 2 razy, ponieważ "istnieje możliwość, że dwa lub więcej urządzeń przystąpi do wysyłania danych w tej samej chwili lub zanim sygnał z pierwszego węzła dotrze do drugiego. W takiej sytuacji żadne z nich nie wykryje sygnału nośnego drugiego. W efekcie obydwa urządzenia wysyłając dane w (prawie) tym samym czasie spowodują kolizję w sieci Ethernet."

Aby CSMA/CD działało poprawnie cała ramka musi być wysłana w tych  $5 \mu s$ , czyli wyliczamy jej długość:

$$N = szybkosc\_transmisji \cdot t_p$$

$$N = 200 000 000 \left[ \frac{b}{s} \right] \cdot 5 [\mu s] = 200 \cdot 5 bit = 1000 bit = 125 B$$

Forczo: ocenione na 1.0 / 1.0 (chyba)

Podobno żeby mieć max. należy jeszcze uwzględnić wykrycie kolizji, czyli przesłanie dodatkowych 32 bitów przez stację. Wynik:

$$125 B + 4 B = 129 B$$

## 2.4 Zadanie

### 2.4.1 Treść

Określić maksymalną długość ramki powstającej w wyniku kolizji w sieci pierścieniowej o sumarycznej długości łączny  $L = 6 km$ ,  $n = 50$  stacji i szybkości transmisji równej 25 Mbitów/s, przy zastosowaniu rywalizacyjnego (CSMA / CD) algorytmu dostępu do łączka.

### 2.4.2 Odpowiedź

Treść tego typu oznacza, że ramka musi przejść przez całą sieć pierścieniową (która idzie tylko w 1 kierunku). Po drodze przechodzi ona przez 49 stacji, które dodatkowo opóźniają sygnał. Odległość między sąsiednimi stacjami to  $\frac{6 \text{ [km]}}{50} = 120 \text{ [m]}$ .

Musimy wziąć więc pod uwagę 2 czynniki: opóźnienie na kablu oraz opóźnienie w węzłach.

1. Czas propagacji:  $\frac{L}{\frac{2}{3} \cdot C} = \frac{6 \text{ [km]}}{200 000 000 \left[ \frac{m}{s} \right]} = 30[\mu s]$

2. Rozmiar ramki:  $25 \text{ [Mbit/s]} \cdot 30[\mu s] = 750 \text{ [b]}$

## 3 TCP / IP

### 3.1 Zadanie

#### 3.1.1 Treść

Przedstaw i porównaj dostępne mechanizmy działania protokołu sterowania przypływem w połączeniach warstwy transportowej i w połączeniach warstwy liniowej (np. SDLC, TCP). W jaki sposób odbierająca stacja transportowa może uzyskać czasowe powstrzymanie transmisji przez nadawcę?

#### 3.1.2 Odpowiedź

- **TCP** - kontrolowanie poprzez zmianę rozmiaru okna odbiorcy.
- **SDLC / HDLC** - kontrolowanie przez sygnały typu WACK / ACK w protokole znakowym BSC (RR, RNR).

## 4 Routing

### 4.1 Zadanie

#### 4.1.1 Treść

C	F	D
213	214	214
178	260	259
U 3	U 2	U 2
V 5	V 4	V 4
Z 4	Z 5	Z 5

Opisz zasadę działa algorytmu dystrybucji pakietów Link State. Przedstaw działanie algorytmu w węźle A w przypadku, gdy węzeł ma jako sąsiadów węzły B, C, D, E i F oraz otrzymał w czasie ostatnich 5 ms pakiety z węzłów C, F i D. Przyjąć, że maksymalny czas oczekiwania na kolejne kopie to 10 ms i węzeł nie odebrał w tym czasie innych pakietów.

#### 4.1.2 Odpowiedź

Przykładowy zestaw pakietów zadziała następująco: stacja A wyśle potwierdzenia do węzłów F i D. Do węzłów B, C i E wyśle pakiet o najwyższym numerze sekwencyjnym (bezpieczniejsza odpowiedź). Dlaczego?

Potwierdzenia wysyłane są do węzłów z których przyszły pakiety z najwyższym numerem sekwencyjnym (drugi wiersz). W tym przypadku są to węzły F i D, których *seq* są równe 214. Do sąsiadów, którzy nie otrzymali potwierdzenia (B, C i E) wysyłany jest ten spośród pakietów z najwyższym numerem sekwencyjnym, którego pole Age (trzeci wiersz) jest największe. Dlatego do tych trzech węzłów zostanie wysłany pakiet o największym *seq*.

#### Inna odpowiedź:

"Pakiet pochodzący z węzła H zostanie odrzucony ponieważ jego numer sekwencyjny jest niższy niż numer pochodzący z F i D czyli zawiera stare dane. Do węzła F oraz D zostanie wysłane potwierdzenie, natomiast do węzłów C, E, H zostanie wysłany pakiet pochodzący z węzła F ponieważ posiada on dłuższy czas życia od pakietu z węzła D. Ten pakiet zostanie również uwzględniony w liczeniu nowej tablicy routingu w węźle A" *Forczu: ocenione na 1.0 / 1.0*

*Ale czas życia nie ma tutaj żadnego znaczenia (Skrzewski potwierdza)*

## 4.2 Zadanie

### 4.2.1 Treść

Węzeł W ma za sąsiadów węzły A, B, C, D i E. Po otrzymaniu pakietu Link State węzeł oczekuje 35 ms, następnie przystępuje do dystrybucji pakietu. W chwili  $t_0$  węzeł odebrał pakiet z węzła B, 5 ms później z węzła D, a po 13 ms pakiet z węzła E. Przedstaw przebieg dystrybucji pakietów

E		B		D	
55		55		55	
240		238		239	
U	2	U	1	U	2
V	3	V	2	V	3
Z	6	Z	5	Z	6

### 4.2.2 Odpowiedź

Do węzłów, których pakiety miały największy nr sekwencyjny (2gi wiersz) wysyła potwierdzenie. Do pozostałych wysyła pakiet tego węzła, który miał największy nr sekwencyjny. Age (3ci wiersz) nie gra roli.

- Do węzła D wysła: potwierdzenie.
- Do węzła C wysła: E.
- Do węzła B wysła: potwierdzenie.
- Do węzła E wysła: potwierdzenie.
- Do węzła A wysła: E.

## 4.3 Zadanie

### 4.3.1 Treść

Węzeł Q generuje co 60 sekund pakiety Link State, w chwili  $t_0$  rozesłał pakiet

$$(Q \mid seq = 23321 \mid age = 300 \mid (U|3) \mid (V|2) \mid (Z|5))$$

Po restarcie w chwili  $t_0 + 25 s$  wysłał pakiet

$$(Q \mid seq = 1 \mid age = 300 \mid (T|2) \mid (U|1) \mid (Z|3))$$

Q	
1	
300	
T	2
U	1
Z	3

a potem kolejne. Przedstaw przebieg dystrybucji tej informacji w sieci. Kiedy te zmiany w topologii zostaną uwzględnione w wyznaczaniu tablicy routingu?

### 4.3.2 Odpowiedź

Działanie Link State:

- "Powiedz wszystkim o swoich sąsiadach" - przekazuje pakiety otrzymane od sąsiednich węzłów do sąsiednich węzłów.
- Nr sekwencyjny - jeśli ktoś dostanie pakiet o większym numerze, to go przetwarza (i przesyła dalej), a jeśli nie jest większy, to nie.
- Wiek - czas życia pakietu (w cyklu). Jeśli czas życia upływa to jest usuwany z tablicy routingu. Liczony w sekundach.

### Odpowiedzi:

1. Węzły odrzucają pakiet z niższym numerem sekwencyjnym, więc nowy pakiet będzie uwzględniony po wygaśnięciu ważenia? pierwszego, czyli za  $275 \text{ s}$ .  
*Ocenione na 0.9 / 1.0*
2. *Forczu:* dlaczego ujebanio ten 0.1? Wg mnie dlatego, że kruczek tkwi w słowie "reset". Pakiety są wysyłane co 60 sekund, ale od momentu resetu ( $t_0 + 25 \text{ s}$ ) zaczynamy liczyć od nowa, czyli nowe pakiety są wysyłane w momentach  $85 \text{ s}, 145 \text{ s}, 205 \text{ s}$  itd.  
Dlatego prawidłową odpowiedzią imo jest  $t_0 + 25 + 5 \cdot 60 = 325 \text{ s}$ , czyli dopiero **5ty** pakiet zostanie uwzględniony przez tablicę routingu.

3. *Inna odpowiedź, ocena nienazna:*

W momencie restartu, po 25 sekundach, oba pakiety żyją i ważniejszy jest Pierwszy z nich, o numerze sekwencyjnym  $seq = 23321$ . Dlatego pakiet o  $seq = 1$  zostanie uwzględniony po przedawnieniu  $seq = 23321$ , czyli po:

$$300 - 25 - t_0 [\text{s}] = 275 [\text{s}]$$

a najpierw zostanie wykorzystany pakiet o  $seq = 23321$ .

4. *by Forczu:* tablica po uwzględnieniu pakiet o  $seq = 23321$

Q	
1	
300	
U	1
V	2
Z	3

Nowe  $U = 3$ , czyli koszt jest wyższy od obecnego, więc nie jest uwzględnione.

Nowe  $Z = 5$ , czyli to samo co wyżej.

Nowe  $V = 5$ , tego nie ma w obecnej tablicy, więc po prostu jest dodane.

## 4.4 Zadanie

### 4.4.1 Treść

Węzeł D odebrał / wysłał do chwili  $T_0$  następujące pakiety.

G	B	C	E	D	F	A	H	C	B
9	8	9	8	9	14	10	9	10	7
3	25	7	28	29	21	23	24	23	23
F	1	A	1	C	2	A	1	B	1
H	1	H	2	H	1	E	1	B	3
D	1	C	2	F	1	B	2	E	1
	1		3	4	5	G	1	D	A
	2					6	7	2	H
							8	1	C
								9	2

W czasie  $T_0 + 5$  węzeł D rozpoczął wyznaczanie nowej tablicy routingu. Przedstawić nową tablicę routingu dla węzła D.

#### 4.4.2 Odpowiedź

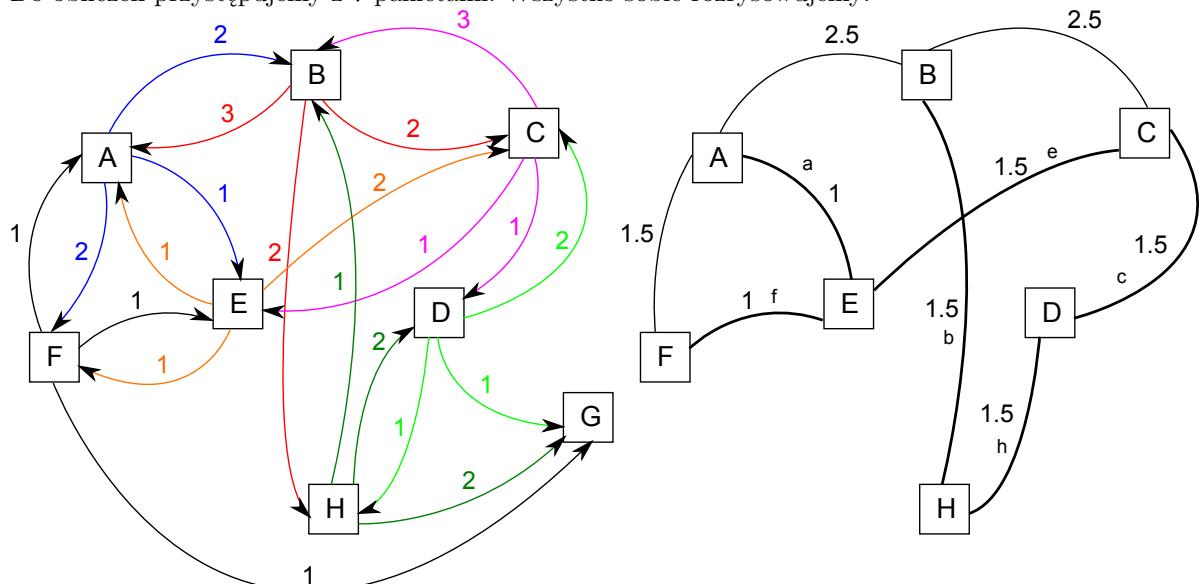
Format pakiety wyraźnie wskazuje, że mamy do czynienia z algorytmem Link State.

- Pierwsza wartość oznacza nadawcę.
  - Druga wartość to numer sekwencyjny - im większy tym pakiet jest ważniejszy.
  - Trzecia wartość to wiek - po tego upływie czasu od momentu odebrania pakiet jest usuwany.

A więc przed rozpoczęciem wyznaczania topologii sieci:

1. Na samym początku **odrzucamy** pakiety nr 1 o czasie życia 3 - w momencie wyznaczania nowej tablicy, w momencie  $T_0 + 5$  już nie istniał.
  2. Następnie patrzymy na numery sekwencyjne - jeżeli odebraliśmy 2+ pakietów od tego samego węzła, to interesuje nas tylko ten o najwyższym numerze sekwencyjnym.
    - Są dwa pakiety od węzła B, nr 2 i 10. Odrzucamy nr 10 o niższym *seq* ( $8 > 7$ ).
    - Są dwa pakiety od węzła C, nr 3 i 9. Odrzucamy nr 3 o niższym *seq* ( $9 < 10$ ).
  3. Odrzucamy te węzły, do których istnieje droga z pakietów innych węzłów, ale owe węzły już nie istnieją, bo zostały odrzucone w dwóch poprzednich krokach. W topologii skutkuje to tylko pojedyncza droga z jednego węzła do drugiego, a interesują nas tylko te z dwiema możliwościami.

Do obliczeń przystępujemy z 7 pakietami. Wszystko sobie rozrysowujemy:



Jeśli węzły da się połączyć na dwa różne sposoby, tworzymy średnią arytmetyczną. Wybieramy najkrótsze możliwe połączenie do każdego wezła, **pomijamy G**, i rysujemy tabelkę.

Cel	Koszt	Przez
A	4	c e a
B	3	h b
C	1.5	c
D	0	-
E	3	c e
F	4	c e f
G	-	-
H	1.5	h

Forcza: ocenione na 1.0 / 1.0

*Podobno uwzględnienie G to 0.0 / 1.0.*

## 4.5 Zadanie

### 4.5.1 Treść

Węzeł K po restarcie wysłał do sąsiadów L, I, F swoją początkową tablicę routingu i otrzymał od nich ich tablice routingu. Wyznacz nową tablicę routingu węzła K.

K	
I	1
F	2
L	1

L	
D	3
C	4
G	2
H	1
N	2
M	5

F	
A	3
B	2
C	1
E	2
G	1
J	2

I	
J	2
H	3
N	3
M	4
E	3

### 4.5.2 Odpowiedź

Do	Przez	Koszt
A	F	5
B	F	4
C	F	3
D	L	4
E	F	4
F	F	2
G	L	3
H	L	2
I	I	1
J	I	3
K	-	-
L	L	1
M	I	5
N	L	3

W tym wszystkim interesują nas tylko drogi lokalnie najbliższe - te z punktu widzenia węzła K i "ofert" dróg sąsiadów. Jeśli dwie drogi mają ten sam koszt to wybór powinien być obojętny. Sąsiedzi (L, I, F) mają się w końcowej tabeli również znajdować.

*Forczu: ocenione na 1.0 / 1.0.*

## 5 Transmisja danych

### 5.1 Zadanie

#### 5.1.1 Treść

Stacja robocza jest połączona z serwerem z prędkością 11 [Mbps] i stopą błędów  $BER = 7.3 \cdot 10^{-6}$ . Oblicz z jaką średnią prędkością mogą być wysyłane informacje z serwera używając wersji standardowej (*plain vanilla, Go Back N*) protokołu TCP. Wynegocjowany segment danych to 768 [B] (ramka w łączu 832 [B]), rozmiar okna to 18.5 [KB], a średni czas RTT to 55 [ms].

#### 5.1.2 Odpowiedź

- Notacja  $BER = 7.3 \cdot 10^{-6}$  oznacza, że na 1 milion bitów 7.3 z nich są przekłamane. Wyliczamy z nich liczbę przekłamanych bajtów. Dzielimy obie strony przez 7.3, czyli 1 na 136986.3 bitów jest nieprawidłowy. Aby uzyskać bajty, dzielimy oba przez 8, czyli 1 na 17123.3 bajtów jest przekłamany.
- Wyliczamy z tego liczbę przekłamanych ramek:

$$\frac{17123.3 [B]}{832 [B]} \approx 20.58$$

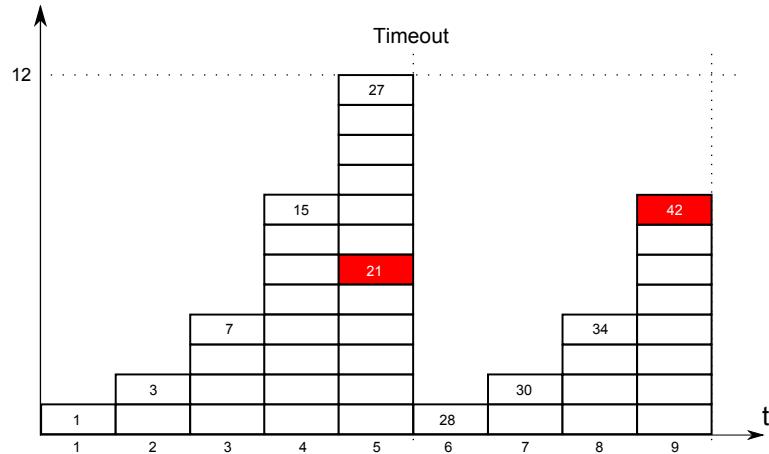
Wybieramy ramkę w łączu, ponieważ interesują nas przesyłane informacje. Wynik oznacza, że 1 z 21 ramek jest przeklamana.

- Teraz liczymy liczbę ramek jaka pomieści się w oknie:

$$\frac{18.5 [kB]}{768 [B]} = \frac{18944 [B]}{768 [B]} \approx 24.67$$

Wybieramy liczbę danych w segmencie, ponieważ interesuje nas to co może pomieścić okno. Z obliczeń wynika, że wielkość okna jest równa 24 ramki + kawałek 25ej, połowa z tego to 12.

- Możemy przystąpić do rysowania słupków:



Liczba ramek rośnie eksponencjalnie aż napotka limit w postaci *slow start thresholdu*. Nie przekracza go. Gdyby w piątej kolumnie nie wystąpił błąd, liczba ramek w kolejnych rosłaby liniowo. Wówczas, po napotkaniu błędu we wzroście liniowym, *threshold* do kolejnego timeoutu byłby mniejszy o połowę.

- Można przystąpić do wyliczenia szybkości:

$$SRET = \frac{(42 - 2) \cdot 832 [B]}{9 \cdot 55 [ms]} \approx 537858.59 \left[ \frac{b}{s} \right] \approx 0.5379 [Mbps]$$

$$GBN = \frac{(42 - 7) \cdot 832 [B]}{9 \cdot 55 [ms]} \approx 470626.26 \left[ \frac{b}{s} \right] \approx 0.47069 [Mbps]$$

## 5.2 Zadanie

### 5.2.1 Treść

Stacja robocza połączona jest z serwerem łączem szeregowym o przepustowości 8 MBit/s, z bitową stopą błędów  $BER = 2 \cdot 10^{-5}$ . Określić z jaką średnią szybkością transmisji danych protokołem TCP w wersji standardowej (*plain vanilla*) można uzyskać jeśli uzgodniona maksymalna wielkość segmentu wynosi 768 B (ramki w łączu 832 B), wielkość danych odbiorcy wynosi 27 kB, czas  $RTT = 40 ms$ .

### 5.2.2 Odpowiedź

Algorytm wykonywania zadań z przekłamaniem:

1. Obliczamy liczbę przekłamań w jednostce czasu jako odwrotność BERu:  $\frac{1}{BER} = \frac{1}{2 \cdot 10^{-5}} = 50000 \rightarrow$  - oznacza to, że 1 na 50k bitów jest przeklamany  $\rightarrow$  co 6250 bajt jest przeklamany.
2. Następnie obliczamy ile ramek zostaje przekłamanych jako:

$$\frac{\text{Odwrotnosc BERu}}{\text{Rozmiar ramki w laczu}}$$

W naszym przypadku jest to  $\frac{6250 [B]}{832 [B]} = 7.51$

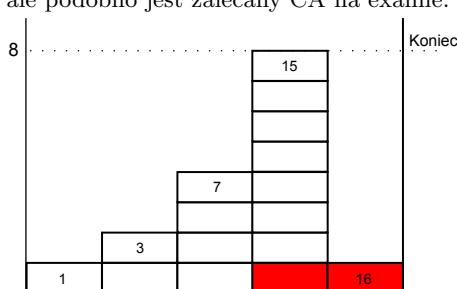
3. Powyższy wynik oznacza, że 7 ramek zostało wysłanych poprawie, a 8 jest przeklamana. Czyli zakładamy, że **co 8 ramka** jest błędna (bierzemy sufit wyniku).
4. Obliczamy liczbę ramek jaką można maksymalnie przesłać w jednym RTT jako:

$$\frac{\text{Rozmiar okna}}{\text{Rozmiar ramki w laczu}}$$

Czyli u nas:  $\frac{27\,000}{832} \approx 32$

5. Następnie stosujemy **metodę słupków** do wyznaczenia właściwego stosunku liczby ramek, jakie można wysłać.

- (a) Rysujemy kolejne transmitowane ramki w postaci słupków złożonych z  $2^n$  ramek: 1, 2, 4, 8 itd.
    - i. Jeżeli protokół jest typu *slow start* - naliczamy bezwzględnie wg  $2^n$  - kolejne słupki mają 16, 32 itd. wysokości
    - ii. Jeżeli protokół jest typu *congestion avoidance* - po przekroczeniu połowy okna zaczynamy zwiększać wysokość **liniowo**, czyli np. dla granicy (*thresholdu*) 8 kolejne słupki będą mieć 9, 10, 11 itd. wysokości.
    - iii. *Congestion avoidance* wykorzystujemy, gdy w poleceniu mamy bufor (podobno).
  - (b) Gdy napotykamy przekłamaną ramkę zaczynamy naliczać od nowa.
  - (c) Liczba ramek w słupku nie może przekroczyć maksymalnej liczby ramek jaką można wysłać w czasie RTT.
  - (d) Kończymy naliczanie gdy przekłamana ramka znajdzie się na szczytce jednego ze słupków, czyli po zakończeniu jednego cyklu.
6. Przystępując do rysowania słupków należy zaznaczyć jaki algorytm wybraliśmy. Niech będzie "slow start", ale podobno jest zalecany CA na examie.



Wynika z tego, że na 16 pakietów 2 zostają przekłamane.  $16 - 2 = 14$  - nieprzekłamane pakiety. Liczba ta nie powinna zostać przekroczena w wysyłaniu danych do okna.

7. Na koniec pozostaje obliczyć efektywną prędkość. Może ona zostać obliczona wg dwóch wzorów:

(a) **SRET**

$$SRET = \frac{((\text{Liczba ramek} - \text{liczba ramek przeklamanych}) \cdot \text{Rozmiar segmentu}}{\text{Liczba kolumn w cyklu} \cdot RTT}$$

(b) **GBN** - Go-Back-N

$$GBN = \frac{((\text{Liczba ramek} - (\text{Ramki przekamane} + \text{Ramki w słupkach ponad nimi})) \cdot \text{Rozmiar segmentu}}{\text{Liczba kolumn w cyklu} \cdot RTT}$$

8. W naszym zadaniu są to:

$$SRET = \frac{(16 - 2) \cdot 768 [B]}{5 \cdot 40 [ms]} = \frac{14 \cdot 768 [B]}{5 \cdot 40 [ms]} = \frac{14 \cdot 768 [B]}{5 \cdot 40 \cdot 10^{-3} [s]} = 53760 [\frac{B}{s}] = 52.5 [\frac{kB}{s}]$$

$$GBN = \frac{(16 - 9) \cdot 768 [B]}{5 \cdot 40 [ms]} = \frac{7 \cdot 768 [B]}{5 \cdot 40 \cdot 10^{-3} [s]} = 26880 [\frac{B}{s}] = 26.25 [\frac{kB}{s}]$$

### 5.3 Zadanie

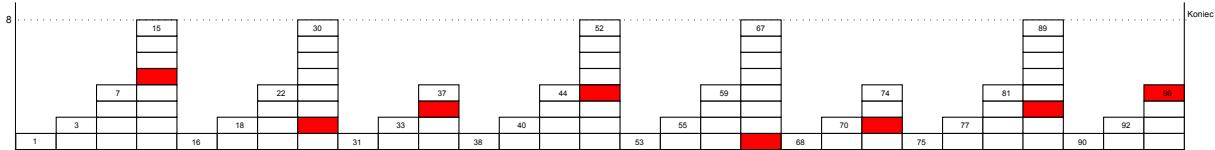
#### 5.3.1 Treść

Stacja robocza połączona jest z serwerem łączem szeregowym o przepustowości  $10 \text{ Mbit/s}$ , z bitową stopą błędów  $BER = 1.3 \cdot 10^{-5}$ . Określ z jaką średnią szybkością można przesyłać dane z serwera protokołem TCP w wersji standardowej, jeśli wielkość segmentu wynosi 768 B (ramki 832 B), wielkość okna odbiorcy 29 kB, średni czas  $RTT = 45 \text{ ms}$ .

#### 5.3.2 Odpowiedź

1.  $\frac{1}{1.3 \cdot 10^{-5}} \approx 76923$  - 1 bit na 76923 jest przekłamany, ok. 1 bajt na 9600 B.
2.  $\frac{9600 [B]}{832 [B]} \approx 11.54$  - co 12 ramka jest przekłamana.
3.  $\frac{29 [kB]}{832 [B]} = \frac{29000 [B]}{832 [B]} \approx 35$  - tyle ramek można maksymalnie przesłać do okna.

Ilustracja przekłamania ramek metodą słupków:



Na 96 pakietów 8 zostaje przekłamanych.

Możemy wyróżnić dwa parametry spełniające zadanie:

1.

$$SRET = \frac{(96 - 8) \cdot 768 [B]}{29 \cdot 45 [ms]} = \frac{88 \cdot 768 [B]}{29 \cdot 45 [ms]}$$

2.

$$GBN = \frac{(96 - 36) \cdot 768 [B]}{29 \cdot 45 [ms]} = \frac{60 \cdot 768 [B]}{29 \cdot 45 [ms]}$$

*Forczo: w tego typu zadaniach przepustowość chyba zawsze jest maksymalna, a protokoół standardowy, co ma upraszczać całość. Ktoś tam napisał, że wielkość ramki we wzorach jest obojętna, ale należy zaznaczyć, czy bierzemy dane czy dane + narzut..*

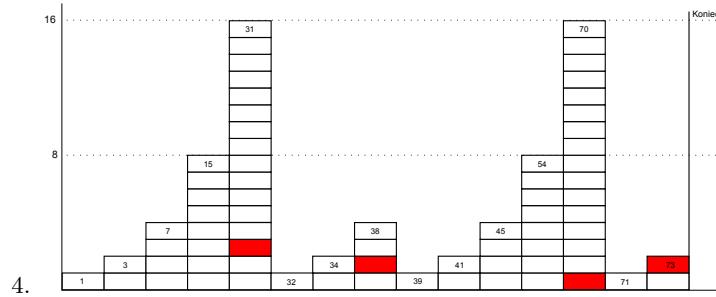
## 5.4 Zadanie

### 5.4.1 Treść

Stacja robocza podłączona jest z serwerem łączem Fast Ethernet o przepustowości  $100 \text{ Mbit/s}$ , z bitową stopą błędów  $BER = 6.6 \cdot 10^{-6}$ . Uzgodniona wielkość segmentu to  $1024 \text{ [B]}$  ( $1088 \text{ [B]}$  w łączu), wielkość okna odbiorcy wynosi  $27 \text{ [kB]}$ , czas RTT wynosi  $40 \text{ [ms]}$ . Określić, ile czasu zajmie przesył  $1 \text{ [MB]}$  tym łączem protokołu TCP w wersji standardowej.

### 5.4.2 Odpowiedź

1.  $\frac{1}{BER} = \frac{1}{6.6 \cdot 10^{-6}} = 151515$ , co oznacza, że 1 na 151515 bitów jest przekłamany  $\rightarrow 1$  na 18939 bajtów.
2.  $\frac{18939 \text{ [B]}}{1088 \text{ [B]}} \approx 17.41 \rightarrow$  co 18 ramka jest przekłamana.
3.  $\frac{27 \text{ [kB]}}{1088 \text{ [B]}} = \frac{27000 \text{ [B]}}{1088 \text{ [B]}} = 24.82 \rightarrow$ , co oznacza maksymalnie 24 ramki w słupku.



4. Odpowiedź

$$SRET = \frac{68 \cdot 1024 \text{ [B]}}{15 \cdot 40 \text{ [ms]}} \approx 113 \left[ \frac{\text{kB}}{\text{s}} \right]$$

$$GNB = \frac{38 \cdot 1024 \text{ [B]}}{15 \cdot 40 \text{ [ms]}} \approx 63 \left[ \frac{\text{kB}}{\text{s}} \right]$$

5. Czas przesyłu 1 MB:

$$1024 \text{ [kB]} : 113 \left[ \frac{\text{kB}}{\text{s}} \right] \approx 9.06 \text{ [s]}$$

$$1024 \text{ [kB]} : 63 \left[ \frac{\text{kB}}{\text{s}} \right] \approx 16.25 \text{ [s]}$$

## 5.5 Zadanie

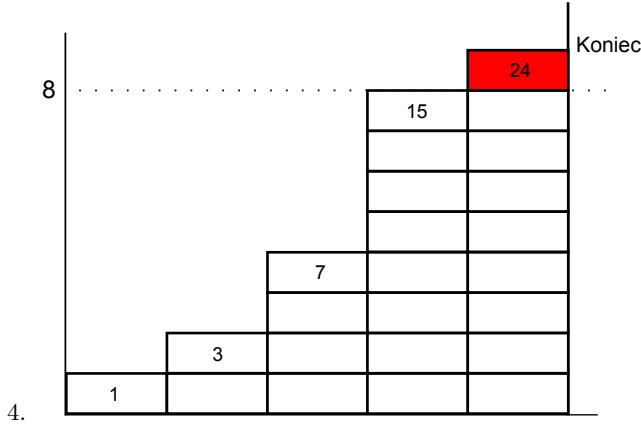
### 5.5.1 Treść

Użytkownik połączony jest z routera łączem sieci lokalnej 10 Mbps Ethernet, czas RTT = 35 ms, router dostępowy na obsługę użytkownika przeznacza bufor 12 kB, rozmiar okna protokołu TCP = 27 kB. Określić maksymalną średnią szybkość transmisji możliwą do uzyskania przez użytkownika przez standardowej wersji protokołu TCP.

### 5.5.2 Odpowiedź

Forczo: ponieważ w treści zadania jest podane, że mamy do czynienia z użytkownikiem łączącym się przez router z Internetem to WYDAJE MI SIE, że należy wykorzystać algorytm congestion avoidance, który ma zastosowanie m.in. w przeglądarkach internetowych. Stąd ta różnica.

1. Ramka w Etherencie: 1500 [B] (1460 [B] dla danych)
2. W routerze jest dostępne 12000 [B].
3.  $\frac{12\ 000 \ [B]}{1500 \ [B]} = 8 \rightarrow$  oznacza, że razem można wysłać 8 ramek, przy 9tej przepłoni się bufor.



4. Na 24 ramki 1 została przekłamana. mamy więc 23 poprawne ramki oraz 5 słupków.
- 5.
- 6.

$$SRET = \frac{23 \cdot 1460 \ [B]}{5 \cdot 35 \ [ms]} = \frac{33580000 \ [B]}{175 \ [s]} = 191885 \left[ \frac{B}{s} \right] = 187 \left[ \frac{kB}{s} \right]$$

## 5.6 Zadanie

### 5.6.1 Treść

Stacja robocza jest połączona z serwerem z prędkością 11 [Mbps] i stopą błędów  $\text{BER} = 7.3 \cdot 10^{-6}$ . Oblicz z jaką średnią prędkością mogą być wysyłane informacje z serwera używając wersji standardowej (*plain vanilla, Go Back N*) protokołu TCP. Wynegocjowany segment danych to 768 [B] (ramka w łączu 832 [B]), rozmiar okna to 18.5 [KB], a średni czas RTT to 55 [ms].

### 5.6.2 Odpowiedź

1.  $\text{BER} = 7.3 \cdot 10^{-6} \rightarrow \approx 136986.30 \text{ [b]} \approx 17123.29 \text{ [B]}$
2.  $\frac{17123.29 \text{ [B]}}{832 \text{ [B]}} = 20.58 \rightarrow \text{co 21 ramka przekłamana.}$   
Ponieważ łączem przesyłamy pełne ramki, więc przekłamanie to dane + wszystkie śmiecie.
3.  $\frac{18.5 \text{ [KB]}}{832 \text{ [B]}} = \frac{18\,944 \text{ [B]}}{832 \text{ [B]}} \approx 22.77$   
Oznacza, że  $\text{SST1} = 23$ .

## 5.7 Zadanie

### 5.7.1 Treść

W systemie transmisji posługującym się protokołem SDLC / HDLC (tryb NRM,  $w=3$ , retransmisyjna grupowa) w łączu ze stopą błędów  $p = 10^{-4}$  przesyłane są dane ramkami z polem danych 256 B, ze stacji nadzędnej do stacji podrzędnej. Przedstaw przebieg transmisji w przesyłce 4.5 kB danych, pomiń nawiązanie / zrywanie połączenia. Ile kosztuje (narzut protokołu) obsługa błędów w transmisji?

### 5.7.2 Odpowiedź

Najpierw liczymy ilość potrzebnych ramek

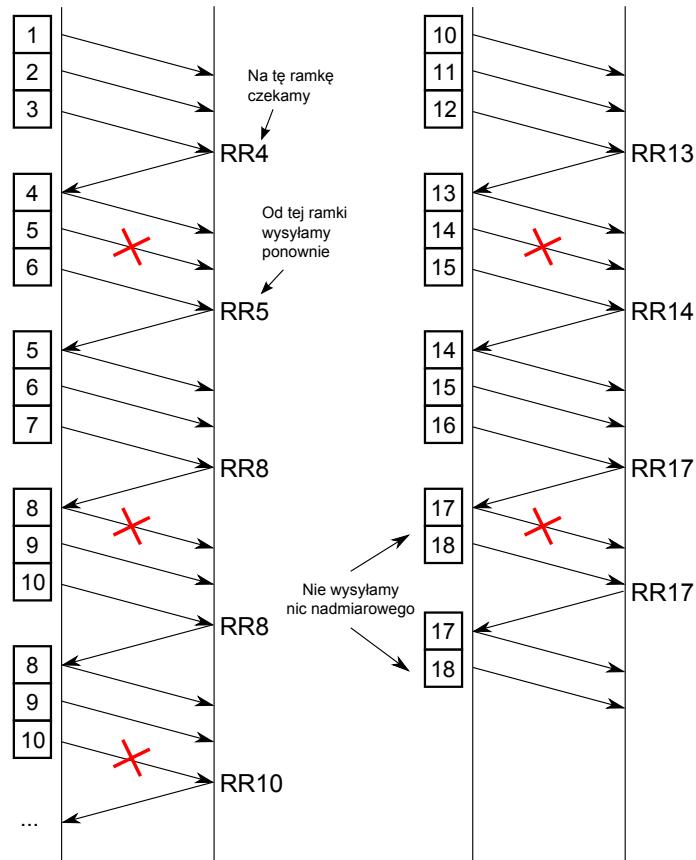
$$\frac{4.5 [kB]}{256 [B]} = \frac{4500 [B]}{256 [B]} \approx 17.58 [\text{ramek}]$$

Należy przesłać 18 ramek.

Następnie należy wyliczyć, co którą ramkę pojawi się przekłamanie. Stopa błędu wynosi 1/10000, czyli to oznacza, że co 10000 bit następuje przekłamanie, co równa się 1250 B. Teraz należy obliczyć, co którą ramkę następuje błąd (i teraz nie jestem pewien ale chyba do tych 256 B danych należy dodać 6B nagłówka).

$$\frac{1250}{262} = 4.77 \approx 5$$

Co piąta ramka zawiera przekłamanie. Teraz można zacząć przedstawiać przebieg transmisji. "w" oznacza ile ramek możemy przesłać w takiej jednej sekwencji.



Następnie należy policzyć narzut (ilość wszystkich ramek - ilość wymaganych ramek)

$$\text{Narzut} = 28 - 18 = 10 \text{ ramek} = 10 * 256 B = 2560 B = 2,5 [kB]$$

Forczu: ocenione na 1.0 / 1.0.

## 5.8 Zadanie

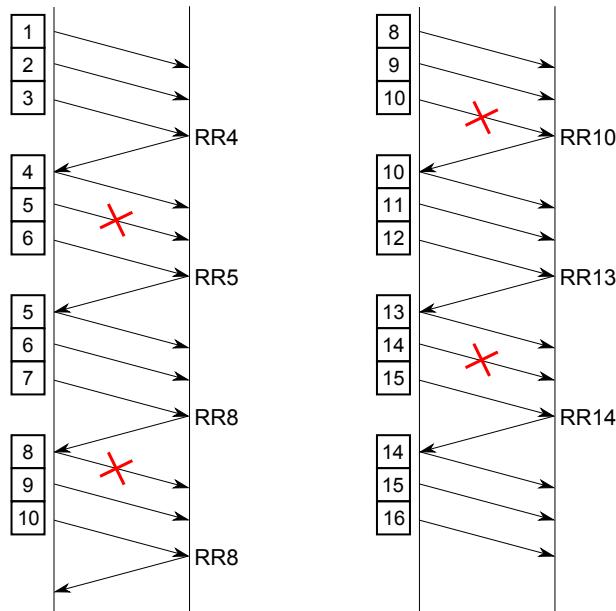
### 5.8.1 Treść

W systemie z protokołem SDLC (tryb NRM,  $w=3$ , retransmisja grupowa), ze stopą błędów  $p = 10^{-4}$ , ramki są wysyłane z polem danych 256 B ze stacji nadzędnej do podrzędnej. Przedstaw przebieg transmisji przy przesyłce 4 kB danych. Pomiń nawiązanie i zerwanie połączenia. Ile kosztuje (narzut protokołu) obsługa błędów w transmisji?

### 5.8.2 Odpowiedź

$BER = p = 10^{-4} \rightarrow 1 \text{ błąd co } 10\,000 \text{ bitów} \rightarrow 1 \text{ błąd do } 1250 \text{ bajtów} \rightarrow 1 \text{ błąd w przybliżeniu co } 5 \text{ ramek.}$

$$\frac{4 [kB]}{256 [B]} = 16 [\text{ramek}]$$



Wysłaliśmy łącznie 24 ramki. Narzut:  $24 - 16 = 8 \text{ ramek} = 2048 \text{ B} = 2 \text{ kB}$

## 5.9 Zadanie

### 5.9.1 Treść

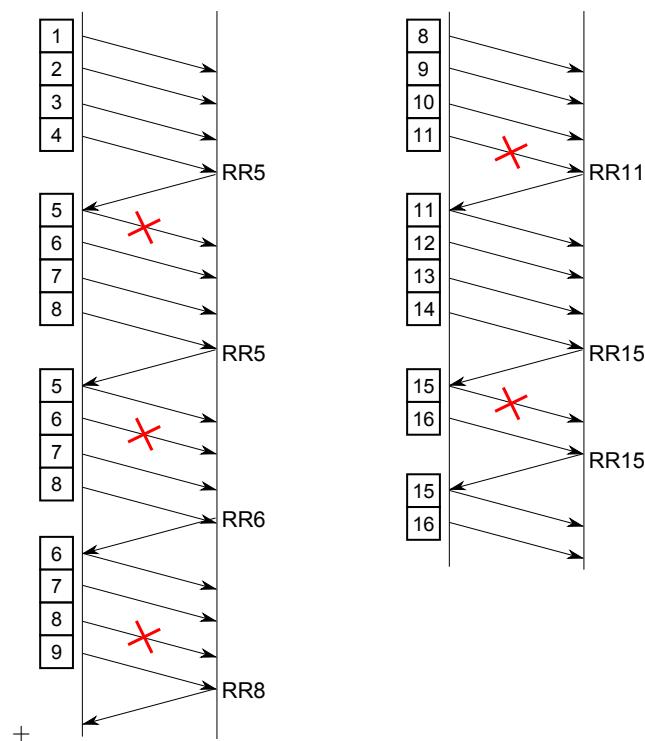
W systemie transmisji z protokołem HDLC średnio jeden na 9000 bitów jest przekłamany. Jaki będzie koszt obsługi błędów transmisji przy przesyłce bloku 4 kB danych ze stacji B do stacji A w tym systemie, jeśli wielkość pola danych to 256 B, okno w=4, tryb NRM z retransmisją grupową.

### 5.9.2 Odpowiedź

$$\frac{4 [kB]}{256 [B]} = 16 \text{ [ramek]}$$

Musimy przesłać 16 ramek.

$\frac{1}{9000}$  bitów przekłamanych  $\rightarrow \frac{1}{1125} [B]$   $\rightarrow$  co 5 ramka przeklamana.



Wysłano 28 ramek. Narzut:  $28 - 16 = 12$  ramek = 3 kB.

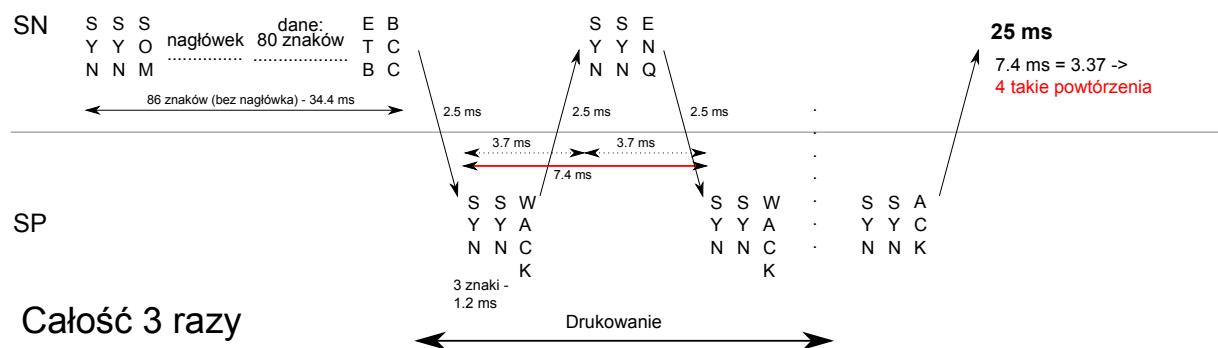
## 5.10 Zadanie

### 5.10.1 Treść

Drukarka wierszowa posiada jeden bufor na 80 znaków (1 wiersz tekstu). Wydruk bufora zajmuje 25 ms, w czasie których drukarka nie przyjmuje danych. Stacja nadzorująca przesyła notatkę - 3 wiersze (bloki po 80 znaków) łączem o szybkości transmisji 19200 bitów / s, protokół BSC (2 znaki SYN). Przedstaw przebieg komunikacji SN-SP (drukarka). Załóż brak przekłamań, pomiń fazę nawiązania połączenia. Przyjmij czas przetwarzania żądania / odpowiedzi i propagacji sygnału łączem 2.5  $\mu$ s.

### 5.10.2 Odpowiedź

Jak mamy bufor to liczymy wg *congestion avoidance*  $19200 \left[ \frac{b}{s} \right] = 2400 \left[ \frac{B}{s} \right]$ , czyli  $2.4k \frac{\text{znaków}}{s}$ , ostatecznie ok. 0.4 ms na jeden znak.



Całość 3 razy

- Sprowadzamy prędkość transmisji do czasu przesyłu jednego znaku: 1 znak / 0.4 ms. Jest to podstawa do większości czasów.
- Stacja nadzorująca, jak się połączy, zaczyna nadawać. Przesyłamy 80 znaków oraz 6 rozkazów. Drukarka odbiera i zaczyna drukować.
- Drukowanie ma trwać (minimum) 25 ms. W tym czasie drukarka wysyła WACKi - SN nie wysyła danych, ale zostaje utrzymywana komunikacja. SN następnie wysyła zapytania ENQ - czy może przesłać następny wiersz.
- Drukarka kończy działanie po 4 wymianach jałowej informacji, po czym wysyła SYN SYN ACK. Po tym sygnale komputer może wysłać kolejne 80 znaków do drukarki. Aby wysłać kolejne 2 wiersze należy całosć powtórzyć jeszcze 2 razy.
- Podobno swietrowi chodziło bardziej o zilustrowanie procesu niż wyliczania czasów.

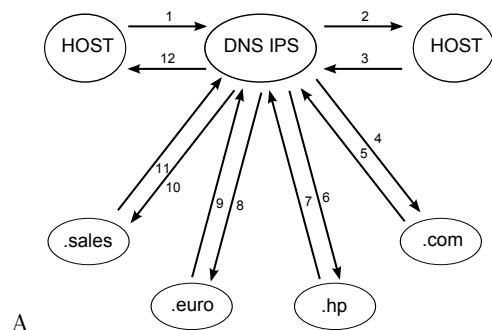
## 6 Adresacja

### 6.1 Zadanie

#### 6.1.1 Treść

Jak przebiega (po raz pierwszy w danych środowisku) proces tłumaczenia nazwy na adres IP dla nazwy daisy.sales.euro.hp.com? Czy zwracany wynik i proces tłumaczenia różni się czymś przy drugiej próbie w stosunku do pierwszej (odstęp między żądaniemi kilka minut, pierwsza zakończona pomyślnie)? Jak będzie przebiegał późniejszy o kilka minut proces tłumaczenia nazwy dino.support.americas.hp.com?

#### 6.1.2 Odpowiedź



- 1 Zapytanie DNS o adres IP dla daisy.sales...
- 2 Jeśli DNS nie posiada tego adresu to pyta root server
- 3 Ten odpowiada, że nie zna adresu IP dla daisy.sales.euro.hp.com , ale zna adres dla com i ... (przekierowuje)
- 4 DNS ISP pyta dns com o daisy.sales.euro.hp.com
- 5 Ten odpowiada, że go nie zna, ale zna adres dla hp
- 6 DNS ISP pyta dns hp o daisy.sales.euro.hp.com
- 7 Ten odpowiada, że go nie zna, ale zna adres dla euro
- 8 DNS ISP pyta dns euro o daisy.sales.euro.hp.com
- 9 Ten odpowiada, że go nie zna, ale zna adres dla sales
- 10 DNS ISP pyta dns sales o daisy.sales.euro.hp.com
- 11 Serwer sales zwraca adres IP domeny daisy.sales.euro.hp.com. Następnie DNS ISP p(...) go do hosta.
- 12 ...

*Brak pełnej treści, ale ocenione na 1.0 / 1.0*

B (a) daisy.sales.euro.hp.com.root

- 1 Zapytanie do NS .com czy "zna" .hp
- 2 Zapytanie do NS .hp czy zna .euro
- 3 Zapytanie do NS .euro czy zna .sales
- 4 Zapytanie do NS .sales czy zna .daisy
- 5 Otrzymanie adresu IP

W drugim przypadku adres hp.com może nie być tłumaczony.

(b) dino.support.americas.hp.com

- 1 Znamy IP (hp.com), pytamy czy zna .americas
- 2 Zapytanie do NS czy zna .support
- 3 Zapytanie do NS czy zna .dino
- 4 Otrzymanie adresu IP

*Ocenione na 1.0 / 1.0*

## 6.2 Zadanie

### 6.2.1 Treść

Dostępna jest pewna ilość kolejnych adresów internetowych, zaczynając od 192.170.0.0. Cztery organizacje A, B, C i D występuły kolejno o przydział puli 1000, 2000, 4000 i 1000 adresów IP. Podaj dla każdej z nich pierwszy i ostatni przypisany adres oraz maskę sieci w notacji w.x.y.z i /m.

### 6.2.2 Odpowiedź

W treści zadania jest podane, że te firmy przychodziły *kolejno*, więc po prostu w tej kolejności przyznajemy im adresy.

1. Firma A, 1000 adresów - maska na 1024 adresy (niezbędne minimum), czyli odcinamy 10 bitów by utworzyć maskę:

$$11111111.11111111.11111100.00000000 = /22 = 255.255.252.0$$

- Pierwszy adres: 192.170.0.0
- Ostatni adres: 192.170.3.255
- w.x.y.z: 255.255.252.0
- Maska: /22

2. Firma B, 2000 adresów - maska na 2048 adresy.

$$11111111.11111111.11111000.00000000 = /21 = 255.255.248.0$$

- Nie możemy w połowie wstawić takiej maski, ponieważ wcześniej utworzyliśmy podsieć na 1024 adresy. Dlatego wymagana jest przerwa do uzupełnienia do 2048. Innymi słowy musimy zostawić przerwę po pierwszej podsieci o długości 1024 adresy, by wyrównać je.
- Pierwszy adres: 192.170.8.0
- Ostatni adres: 192.170.15.255
- w.x.y.z: 255.255.248.0
- Maska: /21

3. Firma C, 4000 adresów - maska na 4096 adresy.

$$11111111.11111111.11110000.00000000 = /20 = 255.255.240.0$$

- Tutaj nie musimy dokonywać analogicznego uzupełnienia. W dwóch poprzednich procesach łącznie przeszliśmy przez 4096 adresów, więc tę sieć możemy wstawić bezpośrednio za drugą.
- Pierwszy adres: 192.170.16.0
- Ostatni adres: 192.170.31.255
- w.x.y.z: 255.255.240.0
- Maska: /20

4. Firma D, 1000 adresów - maska na 1024 adresy

- Mamy wolne miejsce między podsieciami dla firmy A i B, tam dajemy tę sieć.
- Pierwszy adres: 192.170.4.0
- Ostatni adres: 192.170.7.255
- w.x.y.z: 255.255.252.0
- Maska: /22

## 6.3 Zadanie

### 6.3.1 Treść

Firma dysponuje pulą 32 adresów IP. Przyjęto, że pierwszy i ostatni użyteczny adres zostaną przydzielone routero-wi dostępowemu i serverowi DNS. Jeden z komputerów otrzymał adres 172.19.21.137. Podaj elementy konfiguracji IP tego komputera.

### 6.3.2 Odpowiedź

- **Maska podsieci:** 255.255.255.224 (albo /27)  
Firma posiada 32 adresy, które różnią się 5 najmłodszymi bitami.  $32 - 5 = 27$ .
- **Adres IP:** 172.19.21.137
- **Adres sieci:** 172.19.21.128 /27 (to nie jest Host!! za to jest -0.1)  
Adresem sieci wyliczamy zerując 5 ostatnich bitów. (albo ANDując adres z 1.1.1.11100000)
- **Brama / Router:** 172.19.21.129  
Adres routera dostępowego, pierwszy użyteczny adres (00001).
- **Serwer DNS:** 172.19.21.158  
Ostatni użyteczny adres (11110). (Broadcast - 1)
- **Broadcast:** 172.19.21.159  
Adres rozgłoszeniowy, uzyskany przez "zajedynkowanie" 5 ostatnich bitów. Największy adres w sieci.
- Adresy przydzielamy wg kolejnych potęg dwójki  $2^n$ . Gdyby w treści były 33 adresy, to przydzielalibyśmy 64.
- Gdyby w treści należałyby podpiąć 32 klientów, to musielibyśmy mieć 34 adresy i znów przydzielać 64.

## 6.4 Zadanie

### 6.4.1 Treść

Rekord SOA domeny abc.com zawiera następujące wartości:

- refresh = 10800
- retry = 5400
- expire = 86400
- default TTL = 7200

Administrator domeny zaplanował zastąpienie wysłużonego głównego *name servera* nowym systemem (nowy adres IP). Po jakim czasie (od dokonania zmiany rekordu NS) równoleglej pracy obu *name serverów* będzie można wyłączyć stary serwer.

### 6.4.2 Odpowiedź

Po czasie  $TTL + \text{refresh} = 10\,800 + 7\,200 = 18\,000$ . Gdyby się składało z primary i secondary.

## 6.5 Zadanie

### 6.5.1 Treść

Przedstaw jak przebiega proces tłumaczenia adresu IP na nazwę DNS. Jakie domeny TLD są w tym procesie zaangażowane. Ile czasu może maksymalnie zająć przetłumaczenie wcześniej nie znanego adresu, jeżeli odwiedź z NS następują w czasie nie dłuższym niż 0,7s.

### 6.5.2 Odpowiedź

Przykładowy adres: 137.57.25.13

Tłumaczenie: 13.25.57.137.in-addr.arpa

1. NS(arpa)
2. NS(in-addr)
3. NS(137) → zapytanie o główny serwer
4. NS(137.57) → zapytanie o port sieci
5. NS(137.57.25) → zapytanie o podsieć
6. NS(137.57.25.13) → zapytanie o nazwę jednostki w swojej podsieci

Czas trwania tłumaczenia:  $6 \cdot NS = 6 \cdot 0.7 = 4.2 [s]$  *Forczo: należy pomnożyć przez 6: potwierdzone przez Skrzewskiego.*

## 7 Poczta e-mail

### 7.1 Zadanie

#### 7.1.1 Treść

Przedstawić z jakich elementów składa się koperta przesyłki poczty elektronicznej.

- a) Jak przebiega i który z protokołów poczty elektronicznej odpowiada za jej przekazanie?
- b) Jak przebiega odbiór przesyłki przez adresata, który z protokołów obsługuje tę operację?

#### 7.1.2 Odpowiedź

- a) List elektroniczny wysyłany jest za pomocą protokołu SMTP. Klient inicjuje połączenie polecienniem "hello". Następnie podaje adres nadawcy "mail from", adres odbiorcy "rcpt to", treść listu "data". (Koniec części z danymi oznacza pusta linia z '.' - doxus) Wiadomość kończy polecienniem "quit".
- b) Listy odbierane są za pomocą protokołów POP3. Klient żąda listy wiadomości polecienniem "list". Następnie pobiera cały żądany list i usuwa go z serwera : "Retr" i "Dele". Sesja kończy się polecienniem "quit". Oczywiście zanim to wszystko nastąpi musi zostać przeprowadzona autentyfikacja. Klient POP3 zaczyna sesję od podania nazwy użytkownika "user" i hasła "pass".

Ocenione na 1.0 / 1.0

## 7.2 Zadanie

### 7.2.1 Treść

Przesyłka e-mail zawiera fragment tekstu: "płoną góry, płoną lasy...". Przedstaw sposób zakodowania pierwszych słów treści listu zapewniający (dowolnym łączem) poprawny przesył znaków narodowych.

### 7.2.2 Odpowiedź

Zastosowany algorytm to QPE (*Quoted Printable Encoding*). Treść wiadomości:

p=**B3**on=B9 g=**F3**ry, p=**B3**on=**B9** lasy

Gdzie:

- =B3 - dziesiątne 179 - Ł w ASCII
- =B9 - dziesiątne 185 - ą w ASCII
- =F3 - dziesiątne 243 - ó w ASCII

Dziesiątne kody w ASCII były podane w poleceniu, należało je zmienić na zapis szesnastkowy.

Forcju: należy wyraźnie zaznaczyć którą metodą kodowania się posługujemy, inaczej - 0.1

## 7.3 Zadanie

### 7.3.1 Treść

Tekst w języku polskim zawiera średnio 5% znaków narodowych. Porównać efektywność kodowania (zwiększenie objętości) przesyłek e-mailowych dla obu (dostępnych w MIME) metod bezpiecznego przesyłu rozszerzonego zestawu znaków ASCII. Przy jakim procencie znaków narodowych w przesyłanym tekście efektywność obu metod będzie taka sama?

### 7.3.2 Odpowiedź

W treści mamy podane, że interesuje nas przypadek dla 5% znaków narodowych. Innymi słowy, na 100 znaków mamy 5 polskich.

1. **QPE**: 1 znak polski = 3 zwykłe znaki. Oznacza to, że nadmiarowość QPE wynosi **200%**.

2. **Base64**: koduje **każdy** zbiór trzech bajtów w cztery. Czyli nadmiarowość wynosi  $\frac{1}{3} \cdot 100\%$

Oznacza to, że dla 5% znaków w  $n$  znakowej wiadomości obie metody będą miały efektywności:

$$1. \text{ QPE: } n + 5\% \cdot 200\% \cdot n = n + \frac{5}{100} \cdot \frac{200}{100} \cdot n = n + \frac{1}{10} \cdot n = \frac{11}{10} \cdot n$$

$$2. \text{ Base64: } n + \frac{1}{3} \cdot 100\% \cdot n = \frac{4}{3} \cdot n$$

Wynika z tego, że dla 5% znaków narodowych Base64 zawsze da więcej nadmiaru.

**Dla ilu % znaków narodowych QPE i Base64 będą sobie równe w efektywności?**

$$1. \text{ QPE: } n + p\% \cdot 200\% \cdot n = n + \frac{p}{100} \cdot \frac{200}{100} \cdot n = n \cdot \left(1 + \frac{2 \cdot p}{100}\right)$$

$$2. \text{ Base64: } n + \frac{1}{3} \cdot 100\% \cdot n = \frac{4}{3} \cdot n$$

3. **Porównanie:**

$$(a) n \cdot \left(1 + \frac{2 \cdot p}{100}\right) = \frac{4}{3} \cdot n, \text{ dla } n! = 0$$

$$(b) \frac{2 \cdot p}{100} = \frac{1}{3}$$

$$(c) p = \frac{1}{3} \cdot \frac{100}{2} = \frac{50}{3}$$

$$(d) p = 16\frac{2}{3}$$

Odpowiedź: efektywności tych dwóch metod będą równe dla  $16\frac{2}{3}\%$  znaków narodowych.

## **8 Inne**

### **8.1 Zadanie**

#### **8.1.1 Treść**

Posłaniec przewozi nagrane płyty DVD (9 GB informacji) poruszając się ze średnią szybkością 60 km/h. Do jakiej odległości ma on szybkość przesyłu większą niż sieć 10 Mbitów/s.

#### **8.1.2 Odpowiedź**

# Słownik pojęć sieciowych

## 1 Słownik angielsko-polski

### 1.1 Podstawy

1. **Binary Synchronous Communication** - Binarna komunikacja synchroniczna
2. **Confirmation** - Potwierdzenie
3. **Connection** - Połączenie
4. **Destination Address** - Adres przeznaczenia
5. **Digital device** - Jednostka cyfrowa
6. **Encapsulation** - Enkapsulacja
7. **Encryption** - Szyfrowanie
  - (a) **Application** - Warstwa aplikacyjna
  - (b) **Presentation** - Warstwa prezentacyjna
  - (c) **Session** - Warstwa sesji
  - (d) **Transport** - Warstwa transportowa
  - (e) **Network** - Warstwa sieci
  - (f) **Link Access** - Warstwa łącza danych
  - (g) **Physical Access** - Warstwa fizyczna
8. **File transfer** - Transport plików
9. **Finite-State Machine** - Automat stanów skończonych
10. **Flag** - Flaga
11. **Frame** - Ramka
12. **Framing** - Ramkowanie
13. **Header** - Nagłówek
14. **Host** - Stacja
15. **Layer** - Warstwa
16. **Link** - Łącze
17. **Message** - Komunikat
18. **Network** - Sieć
19. **Package** - Pakiet
20. **Party** - Stacja
21. **Preamble** - Preambuła
22. **Receive** - Odbiór
23. **Remote** - Zdalny
24. **Send** - Nadanie
25. **Source Address** - Adres źródłowy
26. **Token** - Żeton

### 1.2 Transfery

1. **Broadcast Transfer** - Transmisja rozgłoszeniowa
2. **Full two-way service** - Pełny Dwukierunkowy Transfer
3. **Half-duplex Transfer** - Jednokierunkowy Transfer Danych
4. **Reliable Data Transfer** - Niezawodny Transfer Danych
5. **Sliding Window** - Przesuwne Okno

### 1.3 Algorytmy

1. **Go-Back-N** - Cofnij się o N

### 1.4 Błędy

1. **Bit Error Rate** - Stopa błędów bitu
2. **Frame Error Rate** - Stopa błędów ramek

### 1.5 Dostęp do łącza (Protokoły MAC)

1. **Open Access** - Swobodny dostęp
2. **Partial / Medium Control Access** - Częściowo kontrolowany dostęp
3. **Full Control Access** - Kontrolowany dostęp
4. **Token Hold Timer (THT)** - Maksymalny czas przetrzymania Tokenu
5. **Not Token Timer (NTT)** - Czas czekania na token

### 1.6 Rodzaje sieci

1. **Cambridge Ring** - Sieć typu Cambridge (chyba)
  - (a) **Spanning Tree Protocol** - Protokół drzewa rozpinanego
2. **Token Ring** - Sieć pierścieniowa

### 1.7 Tryb pracy sieci

1. **Channel switching** - Komutacja kanałów
2. **Information switching** - Komutacja informacji
  - (a) **Information switching** - Komutacja wiadomości
  - (b) **Packet switching** - Komutacja pakietów

### 1.8 Routing

1. **Routing** - Wybór drogi
  - (a) **Source routing** - Routing źródłowy
  - (b) **Static routing** - Algorytm stałego wyboru
  - (c) **Adaptive routing** - Algorytm adaptacyjny
    - i. **Back learning** - Uczenie zwrotne
    - ii. **Cooperation routing** - Algorytm kooperacji
2. **Broadcast** - Zalewanie
3. **Hot potato** - Algorytm gorącego ziemniaka

### 1.9 Obciążenie

1. **Random Error Discard/Detection** - Algorytm wyrzucania losowych pakietów
2. **Error Recovery** - algorytm retransmisji

## 2 Słownik polsko-angielski

### 2.1 Podstawy

1. **Automat stanów skończonych** - Finite-State Machine
2. **Binarna komunikacja synchroniczna** - Binary Synchronous Communication
3. **Adres przeznaczenia** - Destination Address
4. **Adres źródłowy** - Source Address
5. **Enkapsulacja** - Encapsulation
  - (a) **Warstwa aplikacyjna** - Application
  - (b) **Warstwa prezentacyjna** - Presentation
  - (c) **Warstwa sesji** - Session
  - (d) **Warstwa transportowa** - Transport
  - (e) **Warstwa sieci** - Network
  - (f) **Warstwa łącza danych** - Link Access
  - (g) **Warstwa fizyczna** - Physical Access
6. **Flaga** - Flag
7. **Jednostka cyfrowa** - Digital Device
8. **Komunikat** - Message
9. **Łącze** - Link
10. **Nagłówek** - Header
11. **Odbiór** - Receive
12. **Pakiet** - Package
13. **Połączenie** - Connection
14. **Potwierdzenie** - Confirmation
15. **Preambuła** - Preamble
16. **Ramka** - Frame
17. **Ramkowanie** - Framing
18. **Nadanie** - Send
19. **Sieć** - Network
20. **Stacja** - Party
21. **Stacja** - Host
22. **Szyfrowanie** - Encryption
23. **Warstwa** - Layer
24. **Zdalny** - Remote
25. **Żeton** - Token

### 2.2 Transfery

1. **Jednokierunkowy Transfer Danych** - Half-duplex Transfer
2. **Niezawodny Transfer Danych** - Reliable Data Transfer
3. **Pełny Dwukierunkowy Transfer** - Full two-way service
4. **Przesuwne Okno** - Sliding Window
5. **Transmisja rozgłoszeniowa** - Broadcast Transfer

### 2.3 Algorytmy

1. **Cofnij się o N** - Go-Back-N

## **2.4 Dostęp do łącza (Protokoły MAC)**

1. **Swobodny dostęp** - Open Access
2. **Częściowo kontrolowany dostęp** - Partial / Medium Control Access
3. **Kontrolowany dostęp** - Full Control Access
4. **Maksymalny czas przetrzymania Tokenu** - Token Hold Timer (THT)
5. **Czas czekania na token** - Not Token Timer (NTT)

## **2.5 Rodzaje sieci**

1. **Sieć pierścieniowa** - Token Ring
2. **Sieć typu Cambridge (chyba)** - Cambridge Ring
  - (a) **Protokół drzewa rozpinanego** - Spanning Tree Protocol

## **2.6 Tryb pracy sieci**

1. **Komutacja kanałów** - Channel switching
2. **Komutacja informacji** - Information switching
  - (a) **Komutacja wiadomości** - Information switching
  - (b) **Komutacja pakietów** - Packet switching

## **2.7 Obciążenie**

1. **Algorytm wyrzucania losowych pakietów** - Random Error Discard/Detection
2. **algorytm retransmisji** - Error Recovery