Ruting

Mikołaj Leszczuk

Plan wykładu

- **Pojęcie rutera**
- **Algorytm Dijkstry**
- **Algorytm Forda-Fulkersona**
- **Algorytm Dinica**
- **Protokoły warstwy trzeciej (sieciowej)**
  - **Protokół IPX**
  - **Protokół IP**
  - **Protokół BGP-4**

POJĘCIE RUTERA

Warstwa sieciowa (1/2)

- Warstwa sieciowa jako jedyna zna fizyczną topologię sieci
- Rozpoznaje jakie drogi łączą poszczególne komputery (ang. „***routing***”) i decyduje ile informacji należy przesłać, którą drogą
- Jeżeli danych do przesłania jest zbyt wiele, to warstwa sieciowa po prostu je ignoruje
- Ona nie musi zapewniać, pewności transmisji, więc w razie błędu pomija niepoprawne pakiety danych

Warstwa sieciowa (2/2)

- Często w maszynie warstwa sieciowa jest najwyższa, co oznacza, że to urządzenie jest ruterem
- Nie znajdują się w nim żadne użyteczne dla ludzi aplikacje
- Jedyne jego zadanie, to zapewnienie sprawnej łączności między bardzo odległymi punktami sieci
- To właśnie ruter pozwala, bo potrafi odnaleźć najlepszą drogę do jej przekazania

Ruter (1/3)

- **Ruter** (czasem zwany też z ang. „***router***”) to urządzenie sieciowe, które określa następny punkt sieciowy do którego należy skierować pakiet danych (np. datagram IP). Ten proces nazywa się z ang. „***routingiem***” (rutingiem) bądź ***trasowaniem***. Ruting IP odbywa się w warstwie trzeciej modelu OSI
- Ruting jest najczęściej kojarzony z protokołem IP, choć może także zachodzić w sieciach wykorzystujących inne protokoły, np. IPX (sieci Novell)
- Pierwotne rutery z lat sześćdziesiątych były komputerami ogólnego przeznaczenia; chociaż w roli ruterów można używać zwykłych komputerów, nowoczesne szybkie rutery to wysoce wyspecjalizowane urządzenia; zazwyczaj mają wbudowane dodatkowe elementy w celu przyspieszenia typowych czynności, takich jak przekazywanie pakietów

Ruter (2/3)

- Wprowadzono również inne zmiany w celu zwiększenia pewności działania, takie jak zasilanie z baterii, pamięć trwała zamiast magnetycznej. Nowoczesne rutery zaczynają więc przypominać centrale telefoniczne, a obie te technologie coraz bardziej się upodabniają i prawdopodobnie wkrótce się połączą.
- Aby mógł zajść ruting, ruter musi być podłączony przynajmniej do dwóch podsieci (które można okreslić w ramach jednej sieci komputerowej).

Ruter (3/3)

- Szczególnym przypadkiem rutera jest urządzenie z jednym interfejsem sieciowym, które rutuje pomiędzy dwoma lub większą ilością sieci wydzielonych logicznie na tym pojedynczym interfejsie. Dla sieci Ethernet są to VLAN-y (wirtualne sieci lokalne), dla sieci ATM czy Frame Relay kanały PVC/SVC (Permanent Virtual Circuit/Switched Virtual Circuit, stałe bądź komutowane kanały wirtualne).
- Ruter tworzy i utrzymuje tablicę rutingu, która przechowuje ścieżki do konkretnych obszarów sieci i metryki związane z tymi ścieżkami.

Możliwości rutera (1/2)

- **Połączenie sieci lokalnych (LAN) z sieciami rozległymi (WAN)**
- **Połączenie dwóch LANów wybudowanych przy użyciu różnych technik warstwy drugiej, np.: Ethernet i Token Ring**
- **Dopasowanie nagłówków pakietów do odpowiednich segmentów sieci LAN**
- **Wybieranie najlepszej ścieżki dla pakietów**
- **Optymalizacja wydajności sieci**

Możliwości rutera (2/2)

- **Konieczne przy komunikacji komputerów znajdujących się fizycznie w dwóch różnych sieciach lokalnych**
- **Przechowywanie mapy podsieci Internetu**
- **Przekazywanie danych otrzymywanych z jednej podsieci do innych podsieci**
- **Ruter to (np.):**
  - **osobne urządzenie,**
  - **komputer z dwoma kartami sieciowymi i odpowiednim oprogramowaniem**

Ruting (1/3)

- **Inaczej:**
  - **Trasowanie**
  - **routing**
- **Wyznaczanie trasy dla pakietu danych w sieci komputerowej, a następnie wysłanie go tą trasą**
- **Czas życia (ang. *Time To Live*)**
  - **Pakiety przesyłane przez sieć opatrzone są adresem nadawcy i odbiorcy; zadaniem ruterów, jako węzłów pośrednich między nadawcą a odbiorcą, jest przesłanie pakietów do celu po jak najlepszej ścieżce**
  - **Typowy ruter bierze pod uwagę tylko informacje z nagłówka IP, czyli sprawdza tylko informacje z warstwy sieci (trzeciej) modelu OSI; obowiązkiem rutera IP przy przekazywaniu pakietu dalej do celu jest obniżenie o jeden wartości TTL**
  - **Datagram IP, który trafia do routera z wartością 1 w polu TTL zostanie utracony, a do źródła router odsyła datagram ICMP z kodem TTL Exceeded**

Ruting (1/3)

- W **telekomunikacji**, **ruting** (trasowanie, routing) to wyznaczanie trasy dla pakietu danych w sieci komputerowej, a następnie wysłanie go tą trasą.
- Pakiety przesyłane przez sieć opatrzone są adresem nadawcy i odbiorcy; zadaniem ruterów, jako węzłów pośrednich między nadawcą a odbiorcą, jest przesłanie pakietów do celu po jak najlepszej ścieżce; typowy ruter bierze pod uwagę tylko informacje z nagłówka IP, czyli sprawdza tylko informacje z warstwy sieci (trzeciej) modelu OSI; obowiązkiem rutera IP przy przekazywaniu pakietu dalej do celu jest obniżenie o jeden wartości TTL (***Time To Live***, czas życia); datagram IP, który trafia do routera z wartością 1 w polu TTL zostanie utracony, a do źródła router odsyła datagram ICMP z kodem **TTL Exceeded**

Ruting (2/3)

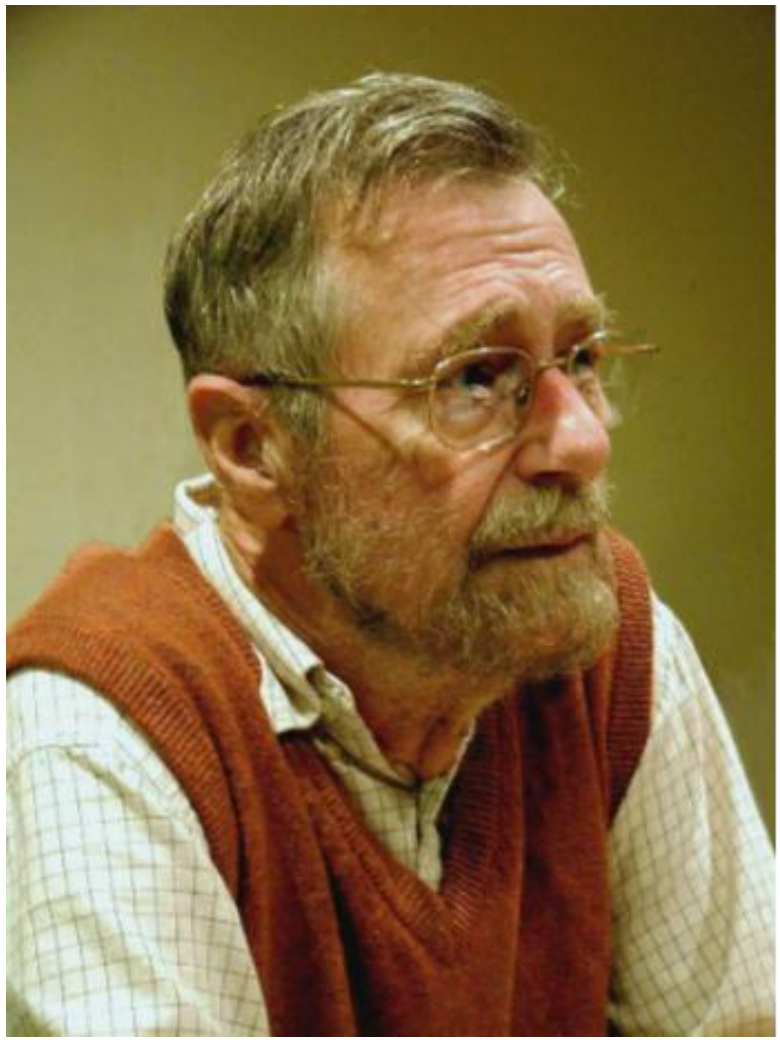
- Rutery utrzymują tablice rutingu, na podstawie których kierują pakiety od określonych nadawców do odbiorców, bądź kolejnych ruterów
- Tablica może być budowana statycznie (**ruting statyczny**) lub dynamicznie (**protokoły rutingu dynamicznego, takie jak** RIP, IGRP, EIGRP, OSPF, BGP, IS-IS)

Ruting (3/3)

- Ruting ma na celu możliwie najlepiej (optymalnie) dostarczyć pakiet do celu; pierwotnie jedynym kryterium wyboru było posiadanie jak najdokładniejszej trasy do celu, ale obecnie protokoły rutingu mogą uwzględniać podczas wyboru trasy również takie parametry jak priorytet pakietu (standardy ToS/DSCP), natężenie ruchu w poszczególnych segmentach sieci itp. W przypadku rutingu brzegowego (wykorzystującego BGP) w Internecie wybór trasy jest silnie związany z polityką poszczególnych dostawców (i zawartymi między nimi umowami o wymianie ruchu) i bywa daleki od optymalnego
- Popularnym algorytmem służącym do wyznaczania tras w sieciach wewnętrznych jest algorytm Dijkstry wyznaczania najkrótszej ścieżki w grafie (np. OSPF)

ALGORYTM DIJKSTRY

Edsger Wybe Dijkstra (11.5.1930 – 6.8.2002).

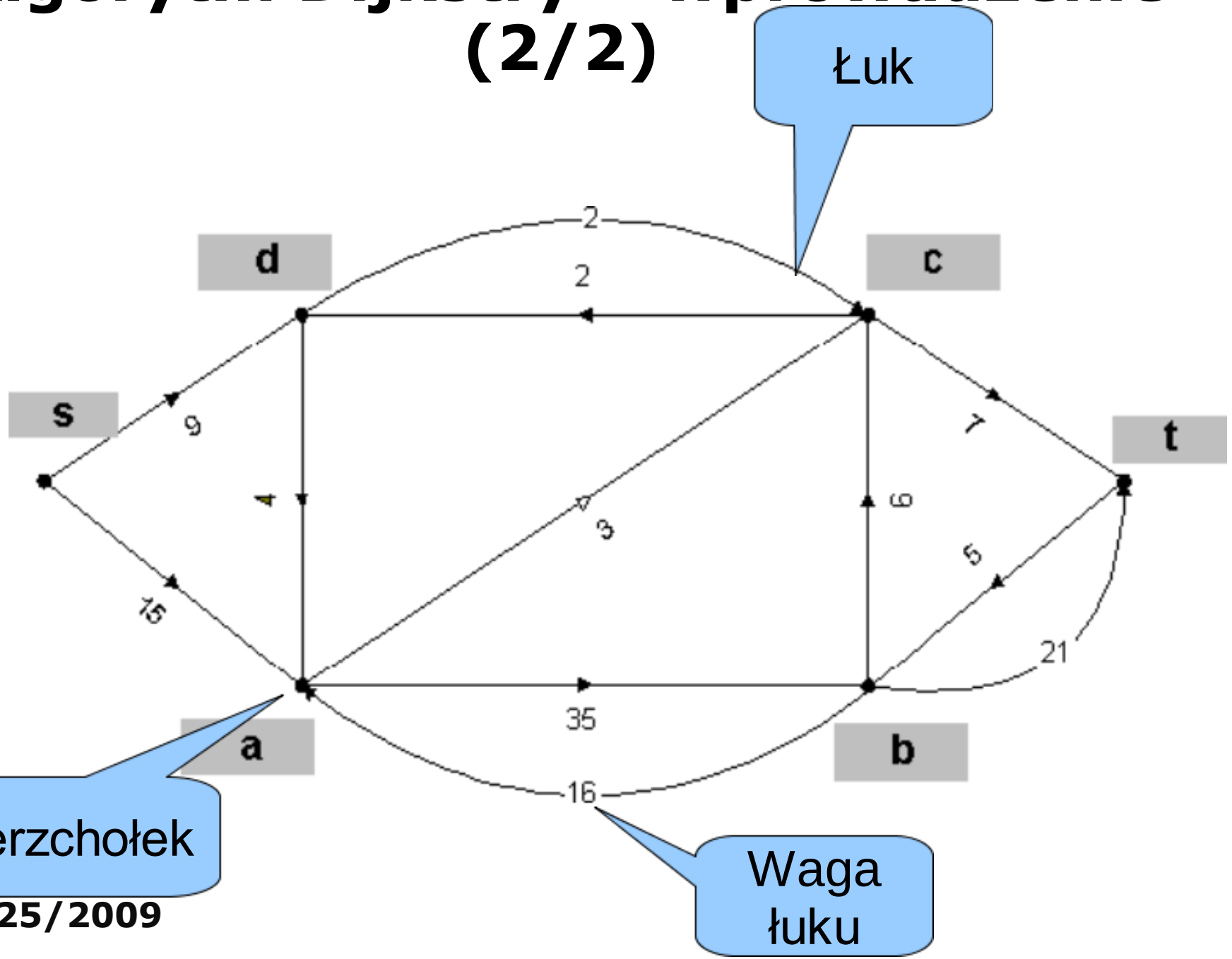


- **Profesor informatyki i matematyki**
- **The university of Texas w Austin**
- **Autor "Algorytmu Dijkstry"**
- **Znajdowanie najkrótszej drogi**
- **pomiędzy dwoma użytkownikami w sieci**
- **zdjęcie: ©2002 Hamilton Richards**

Algorytm Dijkstry – wprowadzenie (1/2)

- **Algorytm Dijkstry jest optymalnym rozwiązaniem problemu najkrótszej drogi w grafie.**
- **Jest też spotykany w literaturze z nieznacznymi modyfikacjami jako algorytm Floyda.**

Algorytm Dijkstry – wprowadzenie (2/2)



Algorytm Dijkstry: Założenia

- **Najkrótszą drogę znajdujemy od wierzchołka początkowego s do wierzchołka ustalonego t.**
- **Wszystkie wagi łuków w sieci są nieujemne.**
- **Przemieszczamy się po łukach sieci z wierzchołka s w kierunku wierzchołka t i cechujemy wierzchołki ich bieżącymi odległościami od wierzchołka s.**
- **Cecha wierzchołka u staje się stała gdy jest równa długości najkrótszej drogi z s do t.**
- **Wierzchołki, które zostały ocechowane stałymi cechami mają cechy tymczasowe.**

Algorytm Dijkstry: Działanie (1/3)

- **Algorytm Dijkstry rozpoczyna działanie od przydzielenia stałej cechy 0 wierzchołkowi s, gdyż 0 jest odległością s od siebie samego.**
- **Wszelkie pozostałe wierzchołki otrzymują cechę tymczasową, gdyż nie zostały dotychczas osiągnięte.**
- **Następnie, każdy bezpośredni następnik v wierzchołka s zostaje oznaczony tymczasową cechą równą wadze łuku (s,v).**

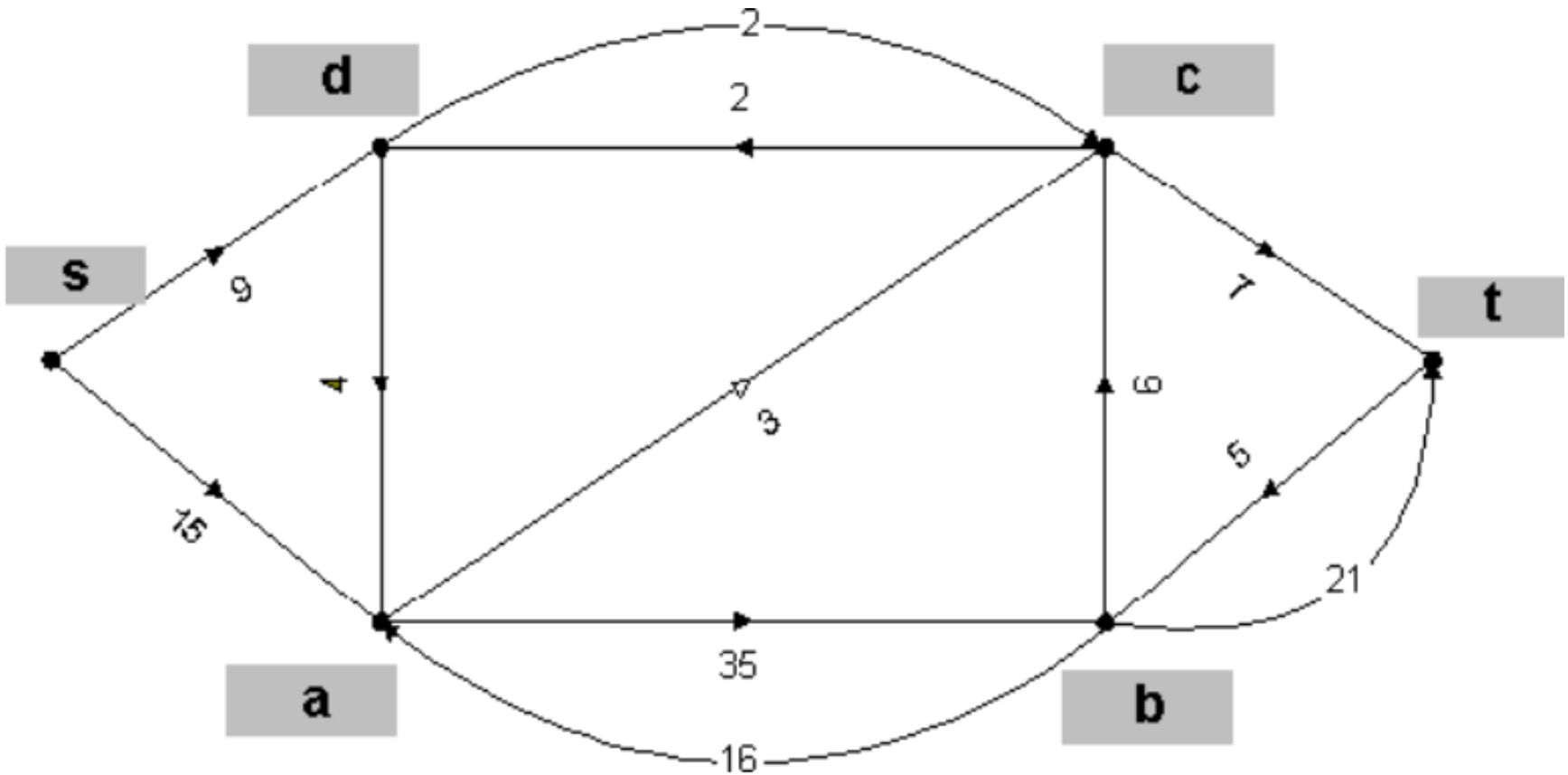
Algorytm Dijkstry: Działanie (2/3).

- Przykładowy wierzchołek x, który ma najmniejszą cechę tymczasową:
  - Jest wierzchołkiem najbliższym wierzchołka s,
  - Bowiem wagi łuków są nieujemne,
  - Nie istnieje droga najkrótsza z s do x.
- Czyli cecha wierzchołka x może zostać ustalona.
- Następnie przeglądamywszystkie bezpośrednie następniki wierzchołka x.
- Zmniejszamy ich cechy tymczasowe:
  - Jeśli droga z s do któregokolwiek z nich,
  - Przechodząca przez x,
  - Jest krótsza od drogi omijającej x.

Algorytm Dijkstry: Działanie (3/3).

- Ponownie znadujemy wierzchołek o najmniejszej cesze tymczasowej, np. y i tę cechę zamieniamy na stałą.
- Wierzchołek y jest drugim najbliższym wierzchołkiem wierzchołka s.
- W taki sposób, w każdej iteracji zmniejszamy wartość cech tymczasowych i zamieniamy na stałą cechę wierzchołka o najmniejszej cesze tymczasowej.
- Konynuujemy to postępowanie, aż do momentu zamiany cechy wierzchołka t z tymczasowej na stałą.

Algorytm Dijkstry: Przykładowa sieć.



Algorytm Dijkstry: Najkrótsza droga z s do t.

	s	a	b	c	d	t
INICJALIZACJA	_0	∞	∞	∞	∞	∞
ITERACJA	_0	15	∞	∞	9	∞
1	_0	15	∞	∞	_9	∞
ITERACJA	_0	13	∞	11	_9	∞
2	_0	13	∞	_11	_9	∞
ITERACJA	_0	13	∞	_11	_9	18
3	_0	_13	∞	_11	_9	18
ITERACJA	_0	_13	48	_11	_9	18
4	_0	_13	48	_11	_9	_18

Algorytm Dijkstry: Zasada działania (1)

- **Na początku wierzchołek s otrzymuje stałą cechę 0, a pozostałe pięć wierzchołków – cechy tymczasowe ∞.**
- **Bezpośrednie następniki wierzchołka s czyli a i d otrzymują nowe cech tymczasowe zredukowane do odpowiednio 15 i 9.**
- **Ponieważ d jest wierzchołkiem o najmniejszej cesze tymczasowej, jego cecha zostaje zmieniona z tymczasowej na stałą.**

Algorytm Dijkstry: Zasada działania (2)

- **Bezpośrednie następniki wierzchołka d czyli a i c otrzymują zredukowane cechy tymczasowe, odpowiednio 13 i 11.**
- **Powtarzamy dalej tak samo.**
- **Kolejne cechy wierzchołków i ich zmianę pokazano w tabeli na wcześniejszym slajdzie.**

ALGORYTM FORDA-FULKERSONA

Algorytm Forda-Fulkersona: Założenia (1/2)

- **Algorytm Forda-Fulkersona służy najogólniej do wyznaczania największego przepływu,
jaki da się uzyskać w niecyklicznym grafie skierowanym.**
- **Funkcjonuje kilka wersji algorytmu wymyślonego w latach 50-tych, w naszym przypadku obiektem rozważań jest wersja z zerowymi warunkami początkowymi.**
- **Mówiąc o warunkach początkowych mamy na myśli istniejące niezerowe przepływy w chwili rozpoczynania działania algorytmu.**

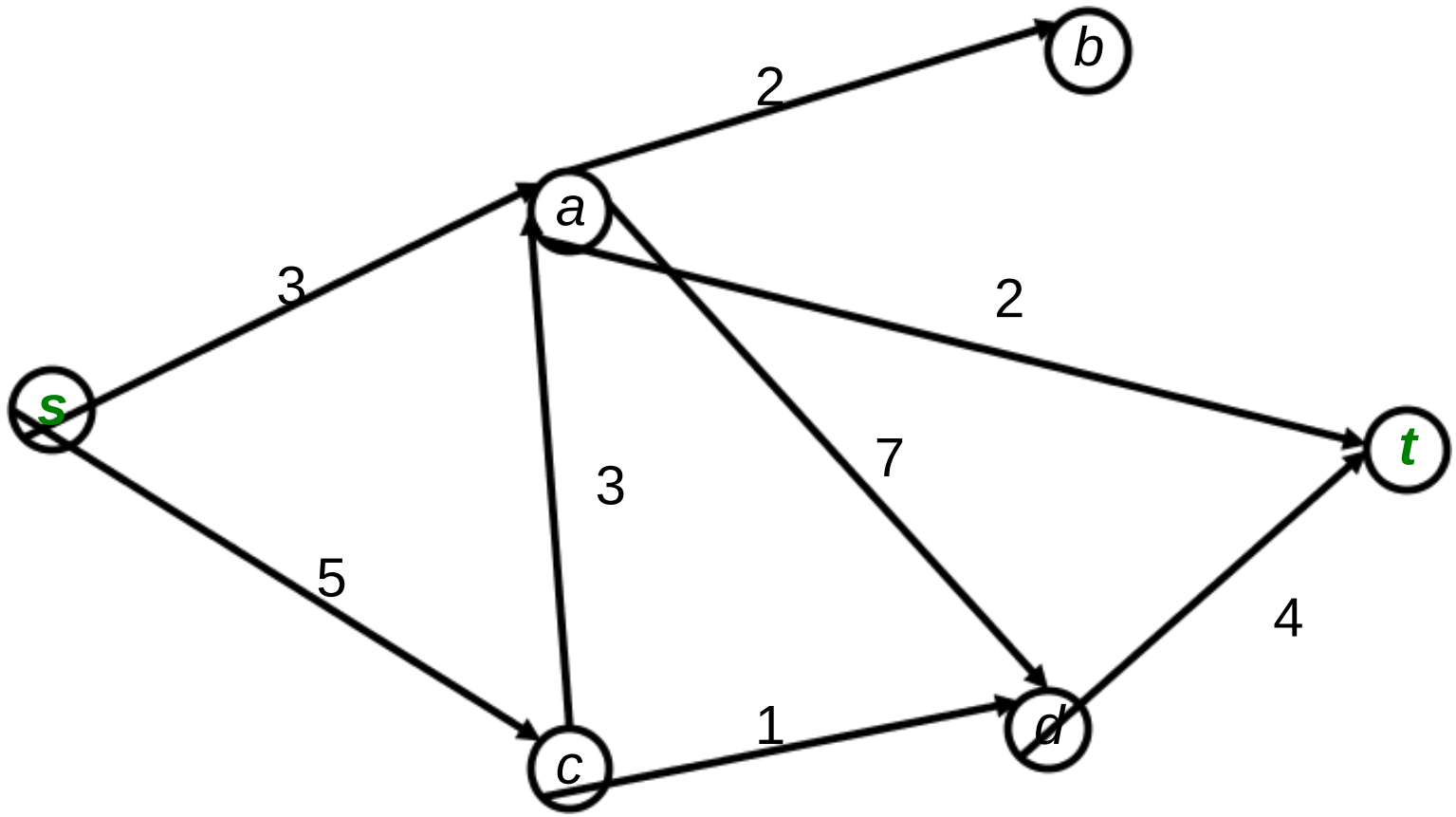
Algorytm Forda-Fulkersona: Założenia (2/2)

- **Wobec takich założeń graf zdefiniowany jest jedynie przez macierz incydencji węzłowych, gdzie elementami macierzy nie są odległości pomiędzy węzłami,**
  jak w przypadku algorytmu Dijikstry, lecz pojemności.
- **Algorytm dysponując tak zadaną macierzą oraz mając wyszczególnione węzły traktowane jako źródło oraz dren oblicza największy przepływ jaki da się uzyskać pomiędzy tymi dwoma węzłami na wszystkich możliwych drogach.**
- **Źródło ma nieskończoną wydajność, przepływ przez graf jest ograniczony jedynie przez pojemności poszczególnych krawędzi grafu.**

Algorytm Forda-Fulkersona: Opis działania algorytmu

- **Działanie algorytmu opiera się na znajdowaniu wszystkich istniejących dróg od źródła s do drenu t przez penetrację grafu w głąb.**
- **Całkowity przepływ pomiędzy źródłem s, a drenem t będzie sumą przepływów na wszystkich możliwych drogach.**
- **Najłatwiej to zademonstrować przy pomocy ilustrowanego przykładu.**

Algorytm Forda-Fulkersona: niecykliczny graf skierowany.



Algorytm Forda-Fulkersona: Macierz incydencji węzłowych

	a	b	c	d	s	t
a	0	2	0	7	0	2
b	0	0	0	0	0	0
c	3	0	0	1	0	0
d	0	0	0	0	0	4
s	3	0	5	0	0	0
t	0	0	0	0	0	0

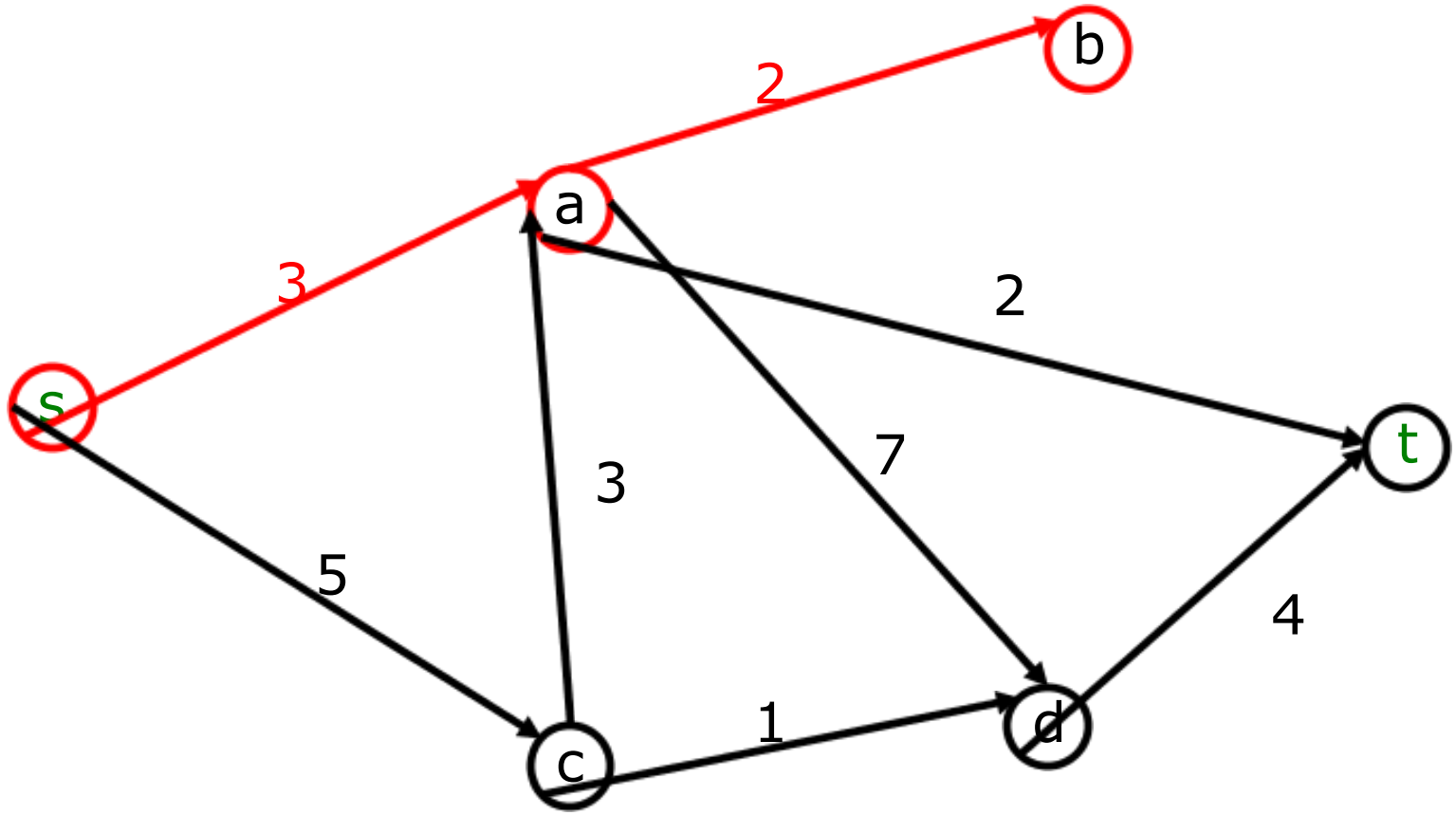
Algorytm Forda-Fulkersona: Wynajdowanie drogi (1)

- ***Każda interesująca nas droga będzie się zaczynała w węźle s, więc w poszukiwaniu węzła po nim następującego przeszukujemy wiersz s poprzedniej macierzy.***
- ***Z węzła s możemy się przemieścić do węzłów oznaczonych a oraz c – w macierzy na przecięciu wiersza s oraz kolumn a oraz c występują niezerowe wartości.***

Algorytm Forda-Fulkersona: Wynajdowanie drogi (2)

- ***Wybieramy pierwszy w kolejności i przemieszczamy się do węzła a.***
- ***Z węzła a stosując tę samą metodę postępowania możemy w analogiczny sposób przemieścić się do węzłów b, d oraz t, wybieramy węzeł b jako pierwszy w kolejności.***

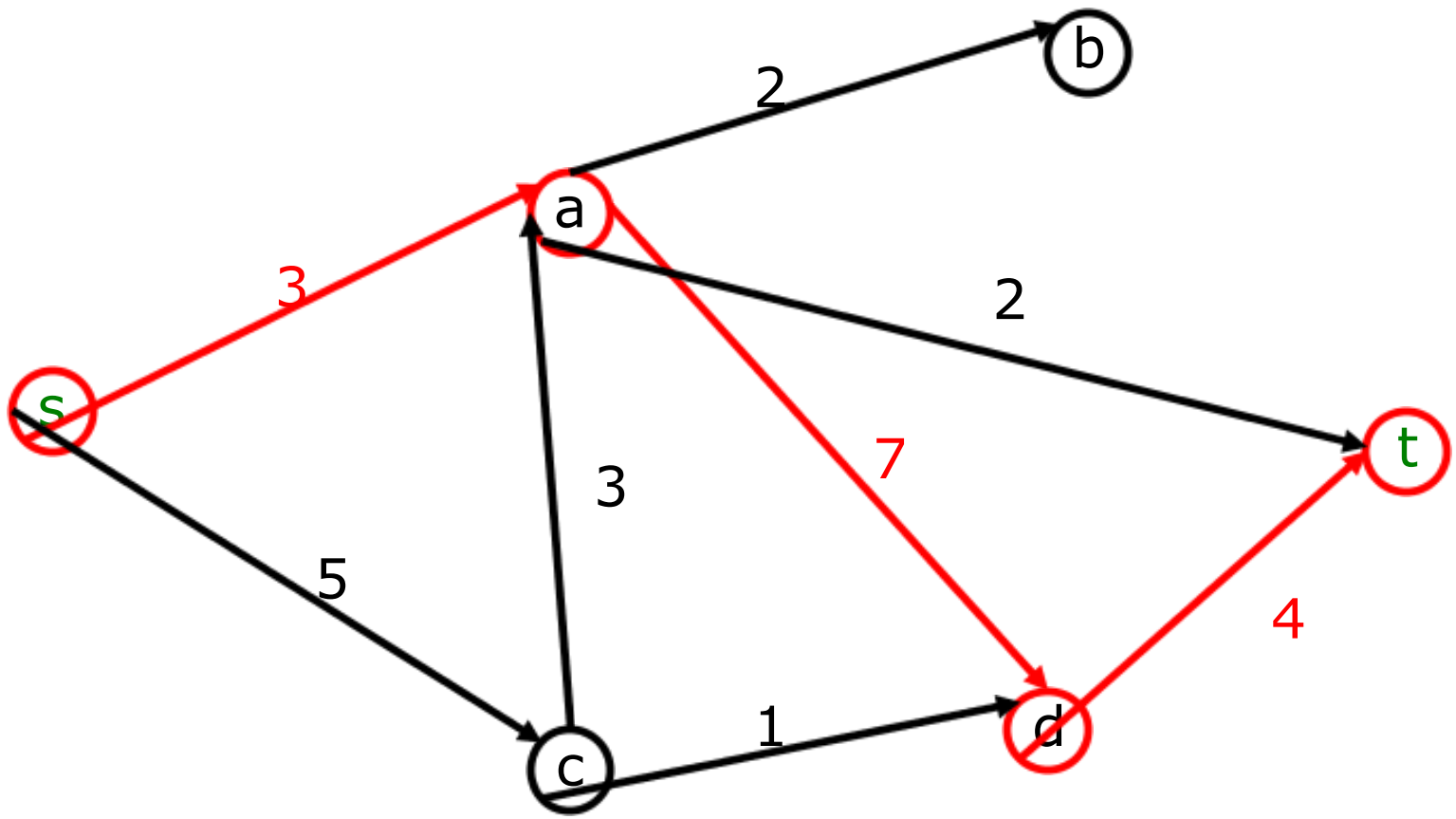
Algorytm Forda-Fulkersona: Wynajdowanie drogi (3)



Algorytm Forda-Fulkersona: Wynajdowanie drogi (4)

- ***Z węzła b nie możemy iść dalej (same zera w wierszu b macierzy), wracamy się więc do węzła a i idziemy do pierwszego po węźle b, czyli do d.***
- ***Z d możemy iść już tylko do t i to kończy nam pierwszą część algorytmu.***

Algorytm Forda-Fulkersona: Wynajdowanie drogi (5)



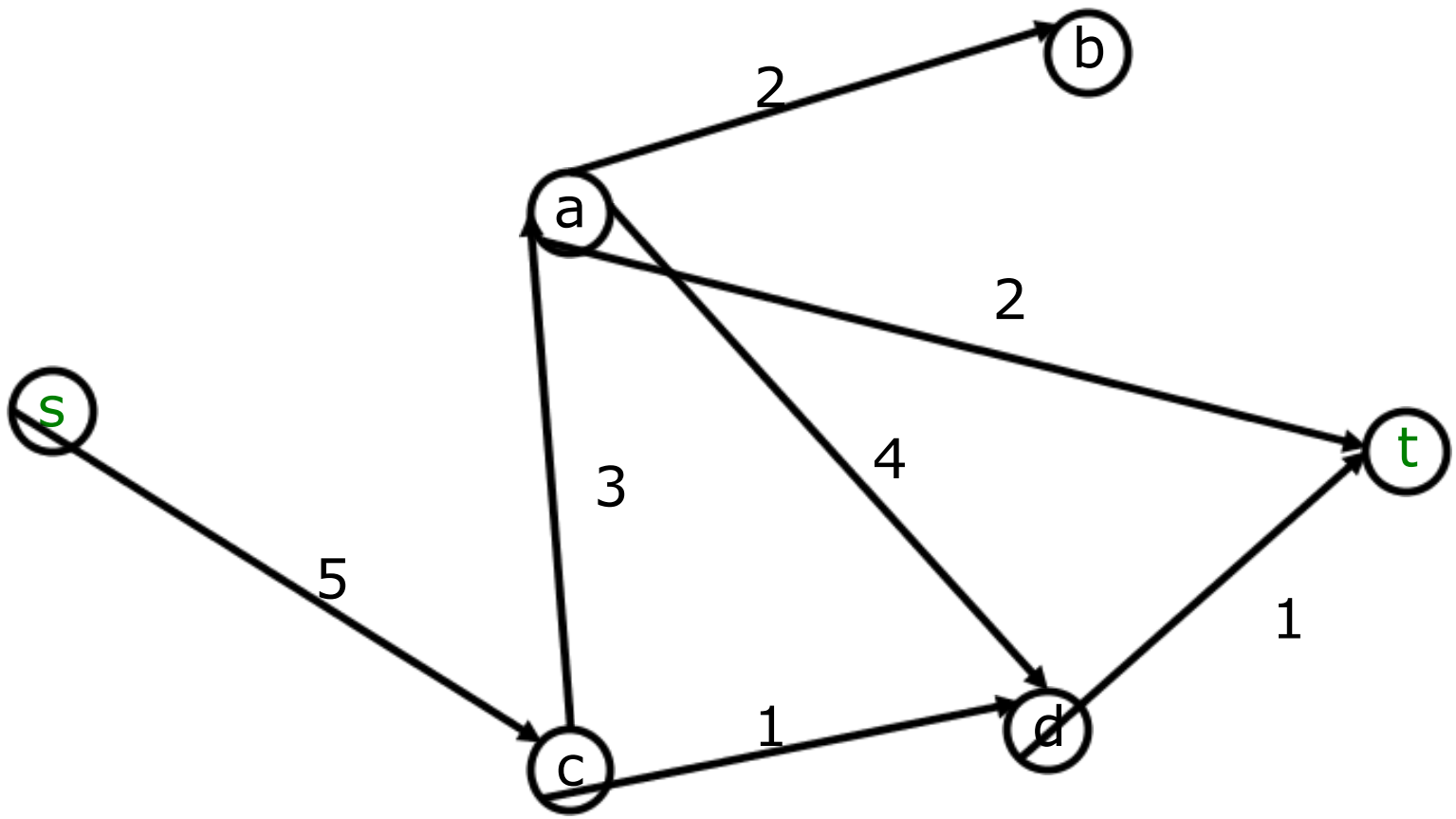
Algorytm Forda-Fulkersona: Wynajdowanie drogi (6)

- **Ważną zasadą w tej części algorytmu jest to, że nawet jeżeli wynika to z przyjętej zasady postępowania, nie przechodzimy do węzła, który występuje w przebytej już drodze!**

Algorytm Forda-Fulkersona: Transformacja grafu

- **Przemieszczając się po drodze (s,a,d,t) znajdujemy krawędź lub krawędzie o najmniejszej pojemności.**
- **W tym przypadku jest to krawędź (s,a) o pojemności równej 3.**
- **Tak więc po wcześniej zdefiniowanej drodze możemy przesłać 3 jednostki i jest to maksymalny przepływ drogi.**
- **Krawędź (s,a) się nasyca i nie uwzględnia się jej w dalszym postępowaniu.**
- **Na pozostałych krawędziach odejmujemy od ich pojemności obliczony wcześniej maksymalny przepływ drogi.**

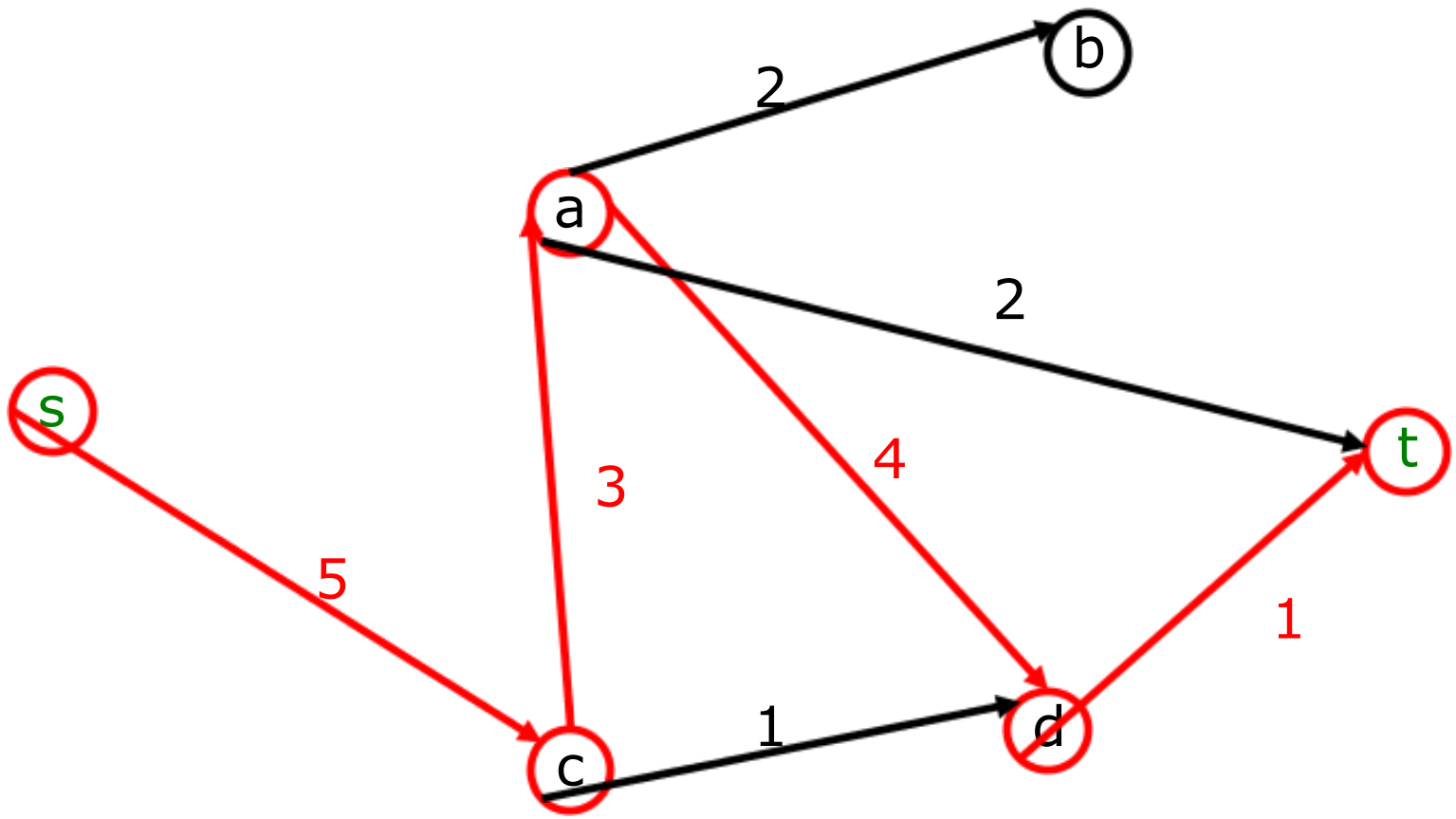
Algorytm Forda-Fulkersona: Graf po transformacji



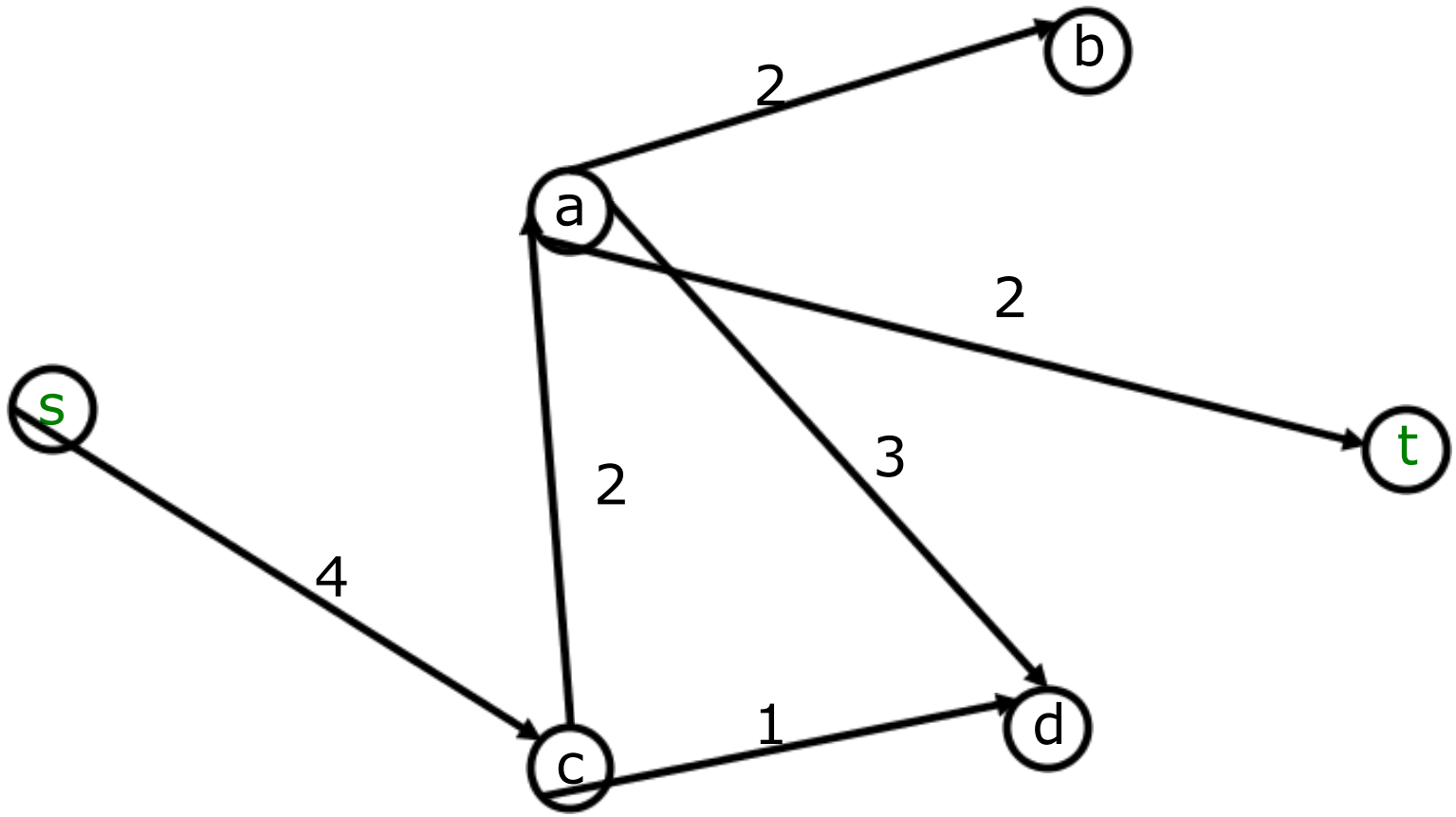
Algorytm Forda-Fulkersona: Na czerwono zmodyfikowane

	a	b	c	d	s	t
a	0	2	0	4	0	2
b	0	0	0	0	0	0
c	3	0	0	1	0	0
d	0	0	0	0	0	1
s	0	0	5	0	0	0
t	0	0	0	0	0	0

Algorytm Forda-Fulkersona: Kolejna dostępna droga



Algorytm Forda-Fulkersona: Po transformacji



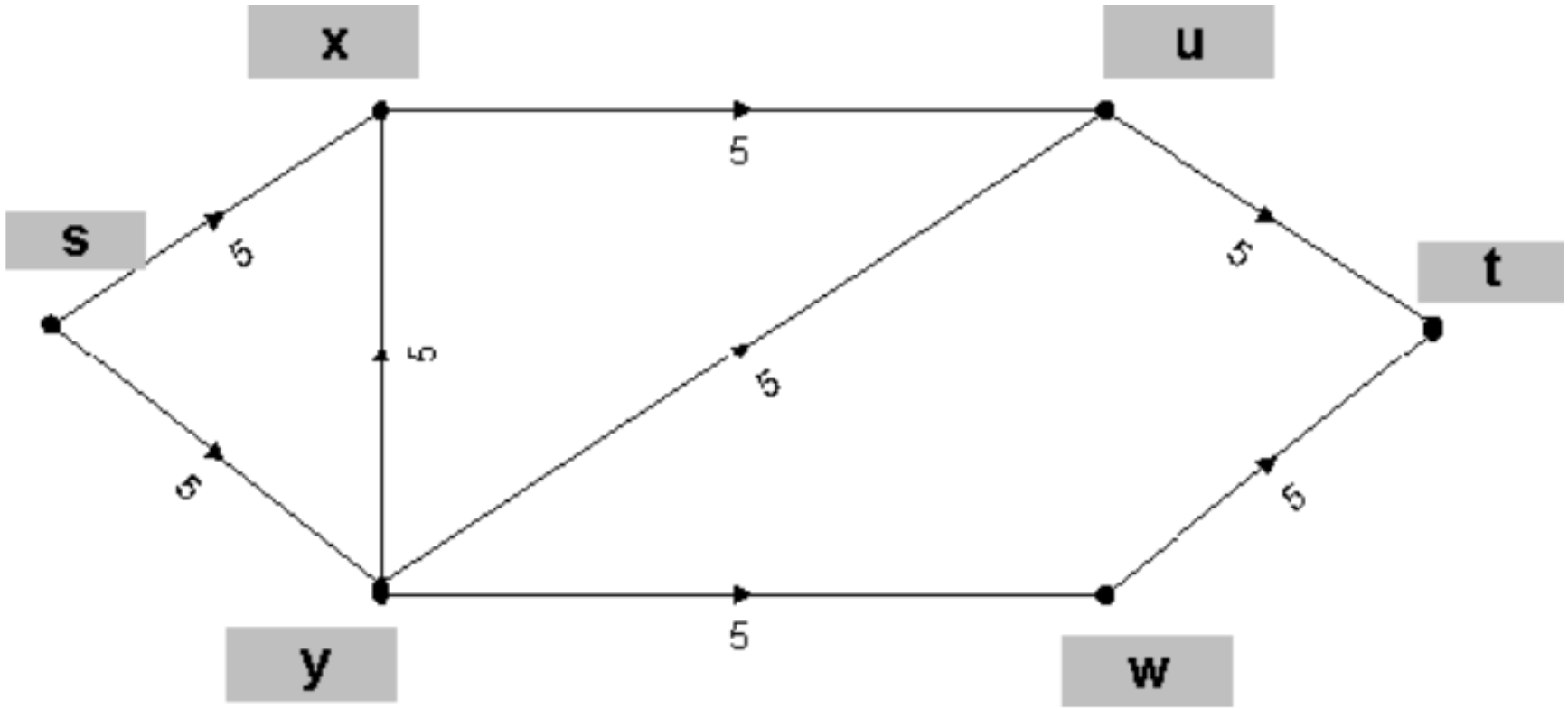
Algorytm Forda-Fulkersona: Po transformacji

	a	b	c	d	s	t
a	0	2	0	3	0	2
b	0	0	0	0	0	0
c	2	0	0	1	0	0
d	0	0	0	0	0	0
s	0	0	4	0	0	0
t	0	0	0	0	0	0

Algorytm Dinica: Założenia

- **Algorytm Dinica w najprostszej wersji wykorzystuje algorytm Forda–Fulkersona.**
- **Modyfikacja polega na tym, że w algorytmie Dinica graf przeszukuje się wszerz a nie wgłąb.**
- **Sam algorytm polega na wielokrotnym powtarzaniu procedury Forda - Fulkersona i modyfikacji grafu.**
- **Przesyłamy pewną ilość towaru ze źródła s stopniowo przez sieć, aż do odpływu t.**
- **Wykrywane są w sieci "użyteczne" i "nienasycone" łuki.**
- **Przesyłmy dalszą ilość towaru.**
- **Takie postępowanie jest kontynuowane aż nic więcej nie może być przesłane z s do t.**

Algorytm Dinica: Przykładowa sieć



Algorytm Dinica: Zasada działania (1)

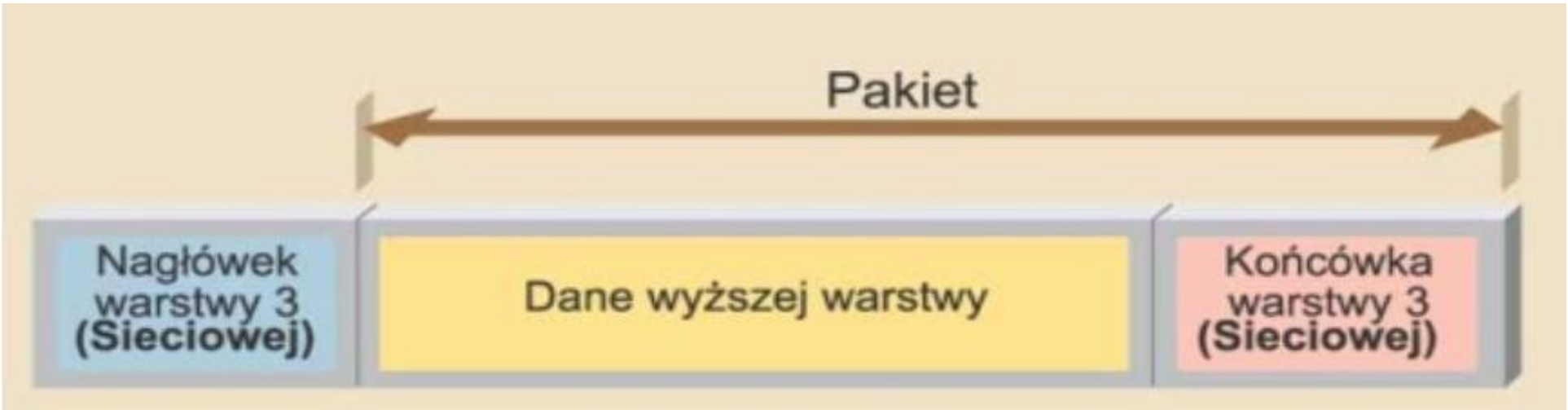
- W naszej przykładowej sieci przepustowość każdego łuku wynosi 5.
- Przykładowo przepuszczamy 5 jednostek towaru z s do y, z y do u i z u do t.
- Na pierwszy rzut oka nie można stwierdzić że w sieci nie ma już drogi z s do t po której można powiększyć przepływ stąd mylne pojęcie o tym iż max. przepływ wynosi 5.
- Jest jednak jeszcze w sieci przepływ z s do t o wartości 10 tzn. 5 jednostek wzdłuż drogi (s,x,u,t) i 5 – po drodze (s,y,w,t).

Algorytm Dinica: Zasada działania (2)

- **Dzięki takiemu podejściu komplikując troszeczkę algorytm wyznaczania maksymalnego przepływu otrzymujemy bardzo korzystne efekty i oszczędzamy dużo potrzebnego i cennego czasu.**
- **Usprawnienie tego algorytmu podali Malhtora, Kumar i Maheshwari.**
- **Polega ono na zastąpieniu algorytmu Forda-Fulkersona inną metodą nasycania krawędzi.**
- **UWAGA! Wagi krawędzi w grafie muszą być liczbami całkowitymi; w przeciwnym razie algorytm Forda-Fulkersona może nie mieć rozwiązania w skończonej liczbie kroków.**

Protokoły warstwy trzeciej (sieciowej)

- Najpopularniejsze: IPX, IP, ATM (ten ostatni to warstwa 2+3)
- Podstawowa jednostka – pakiet, składający się z: nagłówka, danych i końcówki
- Nagłówek i końcówka to informacje sterujące przeznaczone dla warstwy 3 w stacji odbiorczej



Protokół IPX

- IPX = *Internetwork Packet Exchange*
- Steruje:
  - adresowaniem,
  - rutingiem (wyznaczaniem tras) pakietów w:
    - LAN,
    - MAN/WAN.
- Protokół sieciowy sieci Netware
- IPX nie gwarantuje kompletności przesyłanej informacji (możliwe straty pakietów)
- Część tandemu IPX/SPX

Protokół IP
Podstawowe zadania (1/2)

- Opracowany przed amerykański Departament Obrony (Department of Defense), a dokładniej przez Defence Advance Research Projects Agency – DARPA
- Dwie główne wersje:
  - **IPv4 – stosowany obecnie najczęściej**
  - **IPv6 – nowa specyfikacja**

Protokół IP
Podstawowe zadania (2/2)

- Przenoszenie bloków danych zwanych pakietami lub datagramami, ze źródła do celu, gdzie źródła lub cele to zwykle komputery identyfikowane adresami o stałej długości
- Fragmentacja i składanie

Protokół IP
Zakres działania

- Jedynie przenoszenie datagramów poprzez system połączonych sieci
- Brak mechanizmów zapewniających:
  - Integralność danych
  - Kontrolę przepływu
  - Prawidłową kolejność
  - Inne usługi połączeniowe
- Jakość usług (*Quality of Service*):
  - *DiffServ*
  - *IntServ*

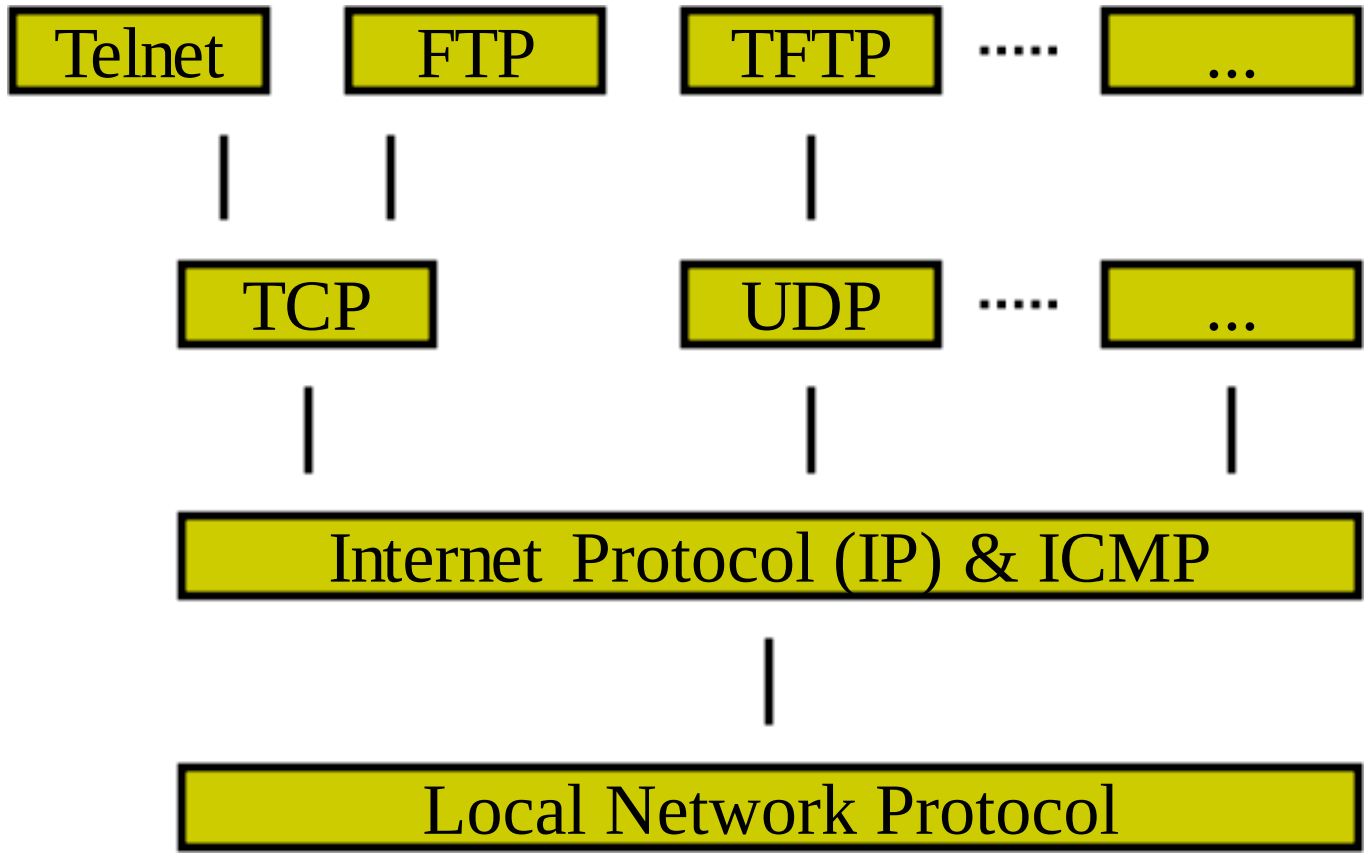
Protokół IP
Podstawowe cechy

- IP = *Internet Protocol*
- Część tandemu TCP/IP
- Odpowiada za:
  - Adresowanie
  - Przesyłanie pakietów IP w sieci
- *Best-effort*
- Protokół bezpołączeniowy

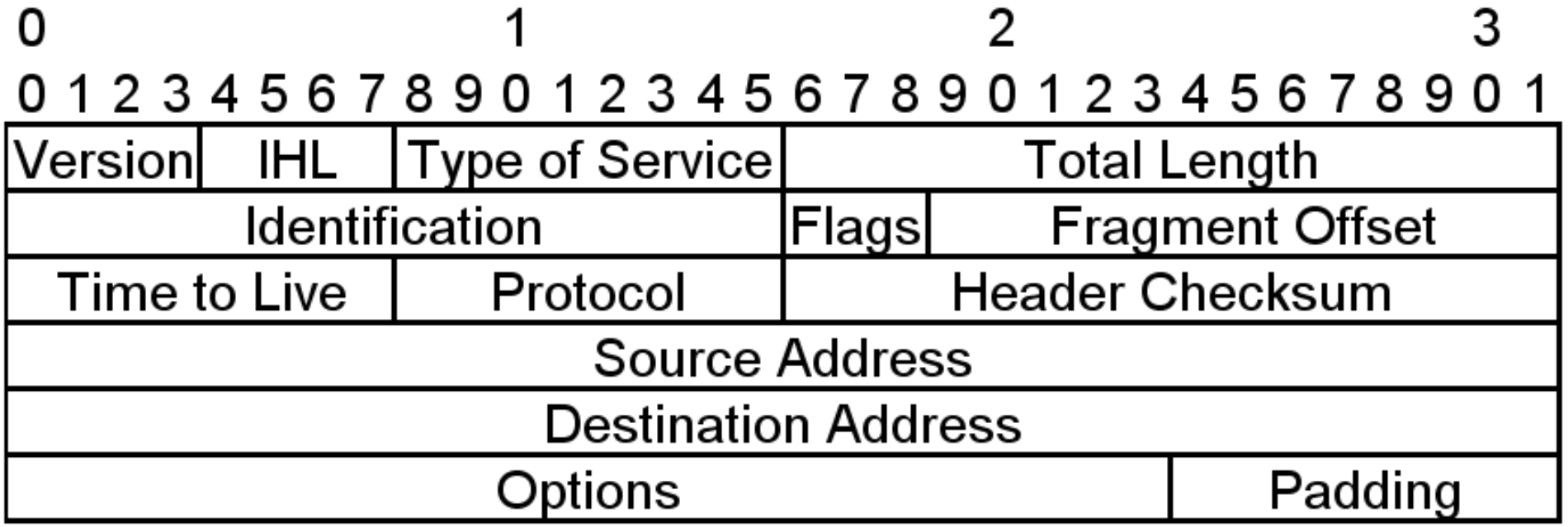
Protokół IP
Interfejsy

- Używany przez protokoły połączeniowe wyższej warstwy
- Protokoły niższej warstwy używane przez IP w celu dostarczania pakietów pomiędzy węzłami
- Przykład 1: protokół TCP wywołujący IP (celem przesłania przez nieg “swoich” danych)
- Przykład 2: protokół IP wywołujący protokół ETHERNET (celem przesłania przez nieg “swoich” danych)

Protokół IP
Współdziałanie protokołów



Protokół IP
Format nagłówka (1/4)



Protokół IP
Format nagłówka (2/4)

- **Version**: **4** bity
  Format nagłówka IP, zwykle wersja **4**
- **IHL**: **4** bity
  Internet Header Length: długość nagłówka IP w **32**-bitowych słowach, czyli wskaźnik na początek danych
- **Type of Service**: **8** bitów
  Parametry żądanej jakości usług, priorytety pakietów (jeśli obsługiwane przez sieć)
- **Total Length**: **16** bitów
  Długość datagramu (pakietu), mierzona w oktetach
- **Identification**: **16** bitów
  Wartość identyfikacyjna nadana przez nadawcę

Protokół IP
Format nagłówka (3/4)

- **Flags**: **3** bity
  Różne flagi sterujące: możliwość (lub nie) fragmentacji, ostatni (lub nie) fragment
- **Fragment Offset**: **13** bitów
  Ustalenie przynależności fragmentu do datagramu, mierzone w jednostkach **8**-oktetowych (**64**-bitowych)
- **Time to Live**: **8** bitów
  Maksymalny czas pozostawania datagramu w systemie, **0=*zniszczenie datagramu***, wartość modyfikowana, mierzona w sekundach, pomniejszana o **1** w węzłach
- **Protocol**: **8** bitów
  Protokół wyższej warstwy użyty do porcjowania danych

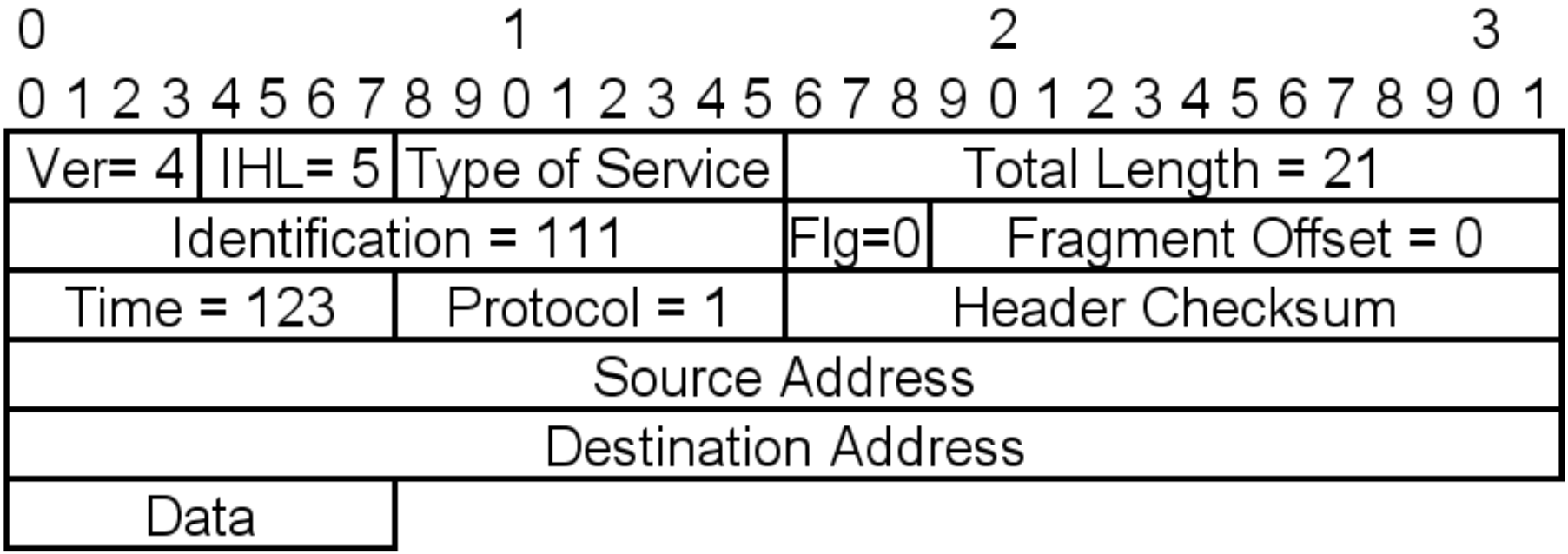
Protokół IP
Format nagłówka (4/4)

- **Header Checksum**: **16** bitów
  Suma kontrolna (wyłącznie nagłówka)
- **Source Address**: **32** bitów
  Adres źródła
- **Destination Address**: **32** bitów
  Adres przeznaczenia
- **Options**: zmienne
  Występowanie w datagramach – opcjonalne, koniecznie implementowane we wszystkich komputerach i ruterach
- **Padding**: zmienne
  Zapewnia podzielność długości nagłówka przez **32** bity, wartość zerowa

Pola modyfikowane przez fragmentację

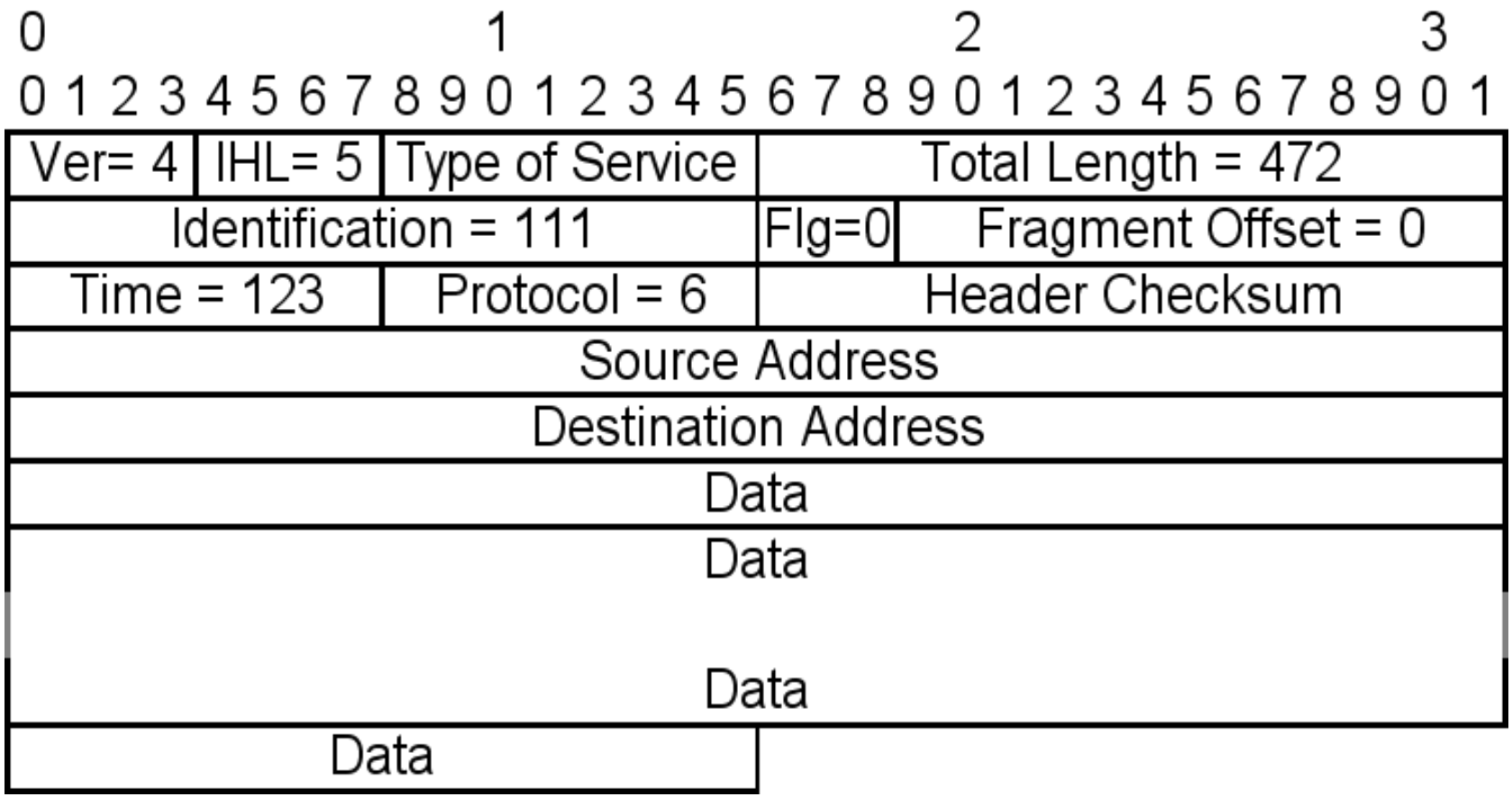
1. Pole opcjonalne
2. Flaga “więcej fragmentów”
3. Przesunięcie fragmentu
4. Pole IHP
5. Pole długości całkowitej
6. Suma kontrolna nagłówka

Przykład 1: Najprostszy pakiet IP

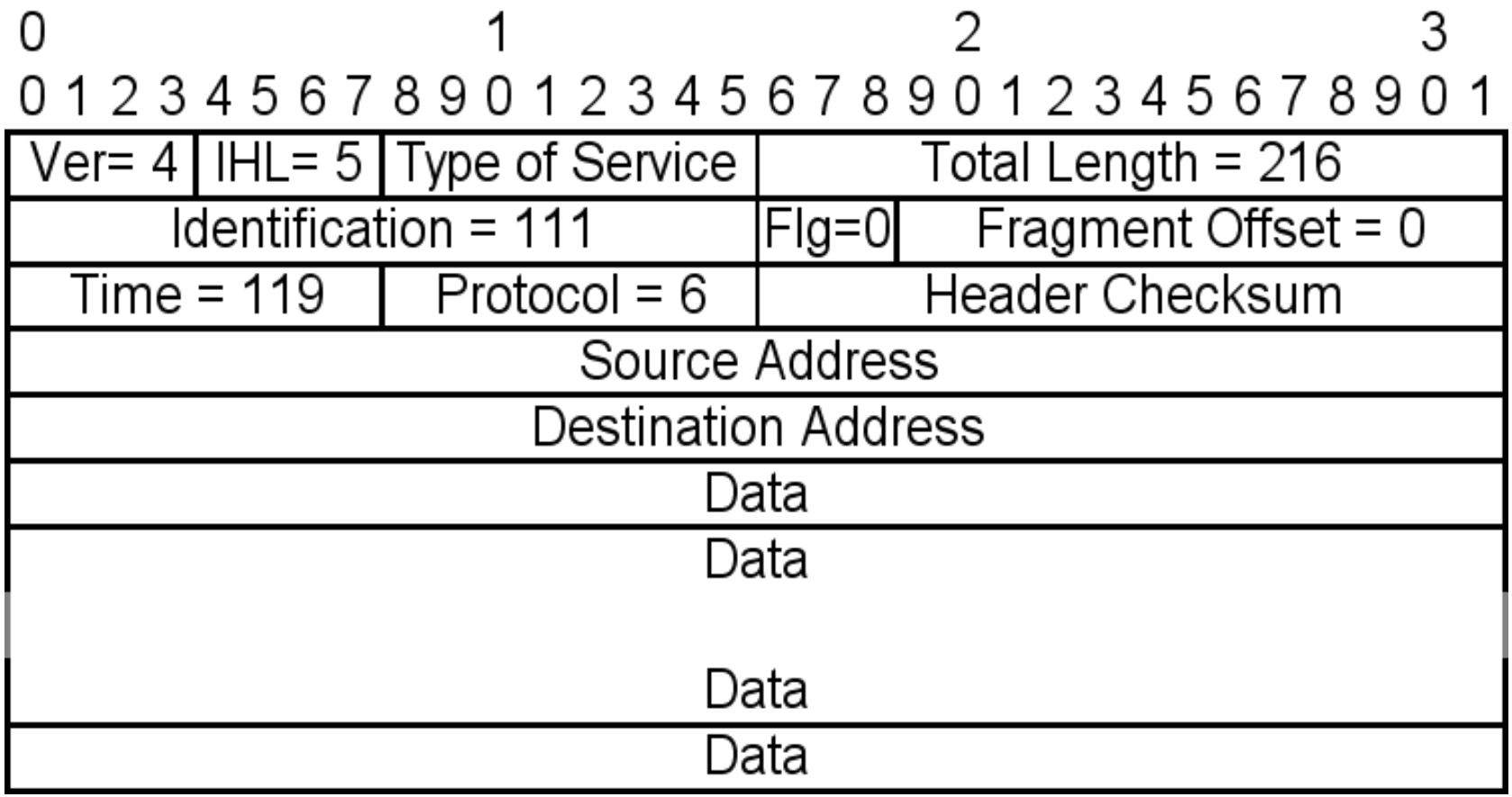


- Datagram IP, wersji **4**
- Pięć 32-bitowych słów nagłówka, długość całkowita datagramu: 21 oktetów
- Datagram kompletny (nie fragment)

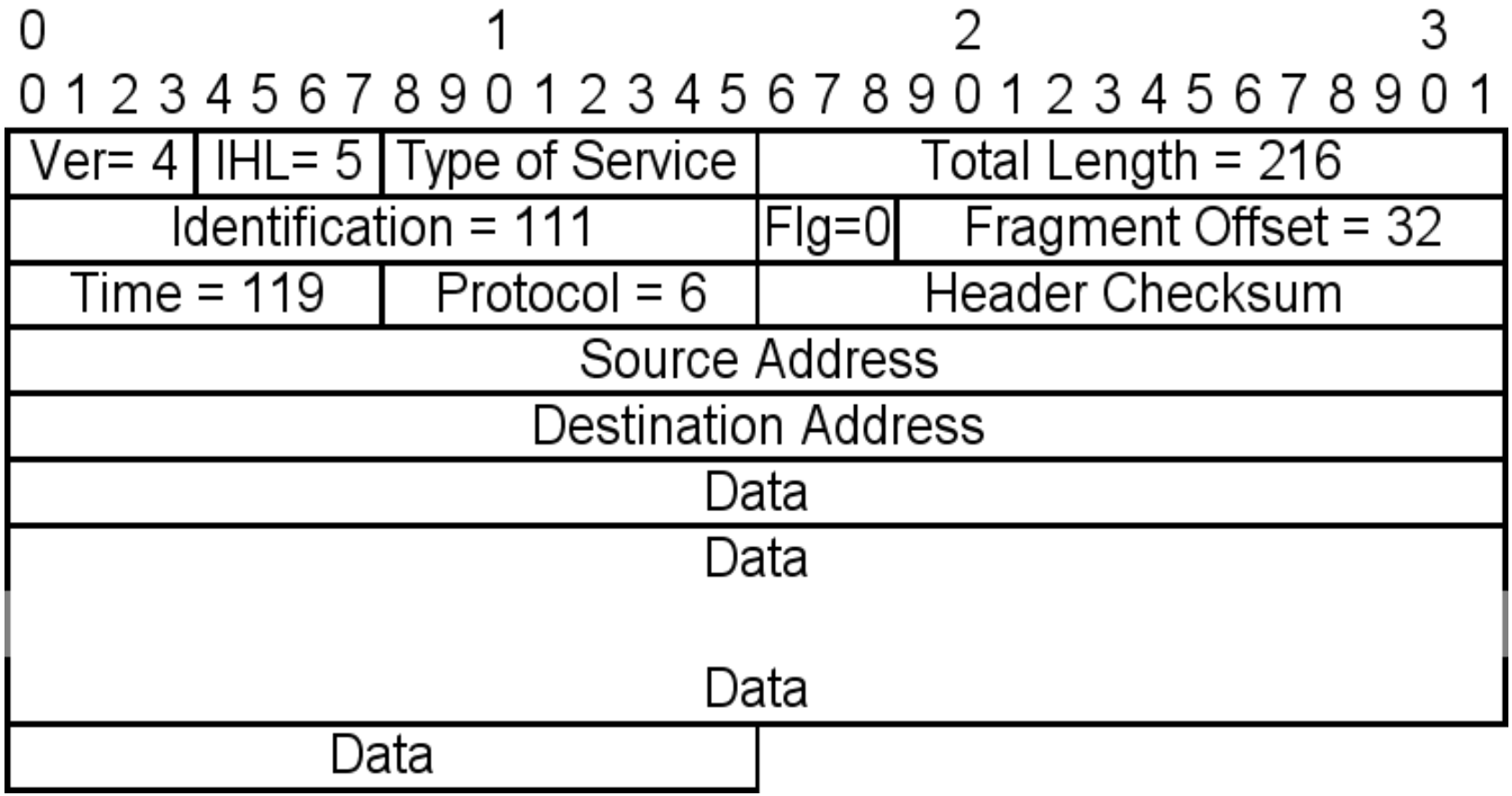
Przykład
2a: Średniej długości pakiet



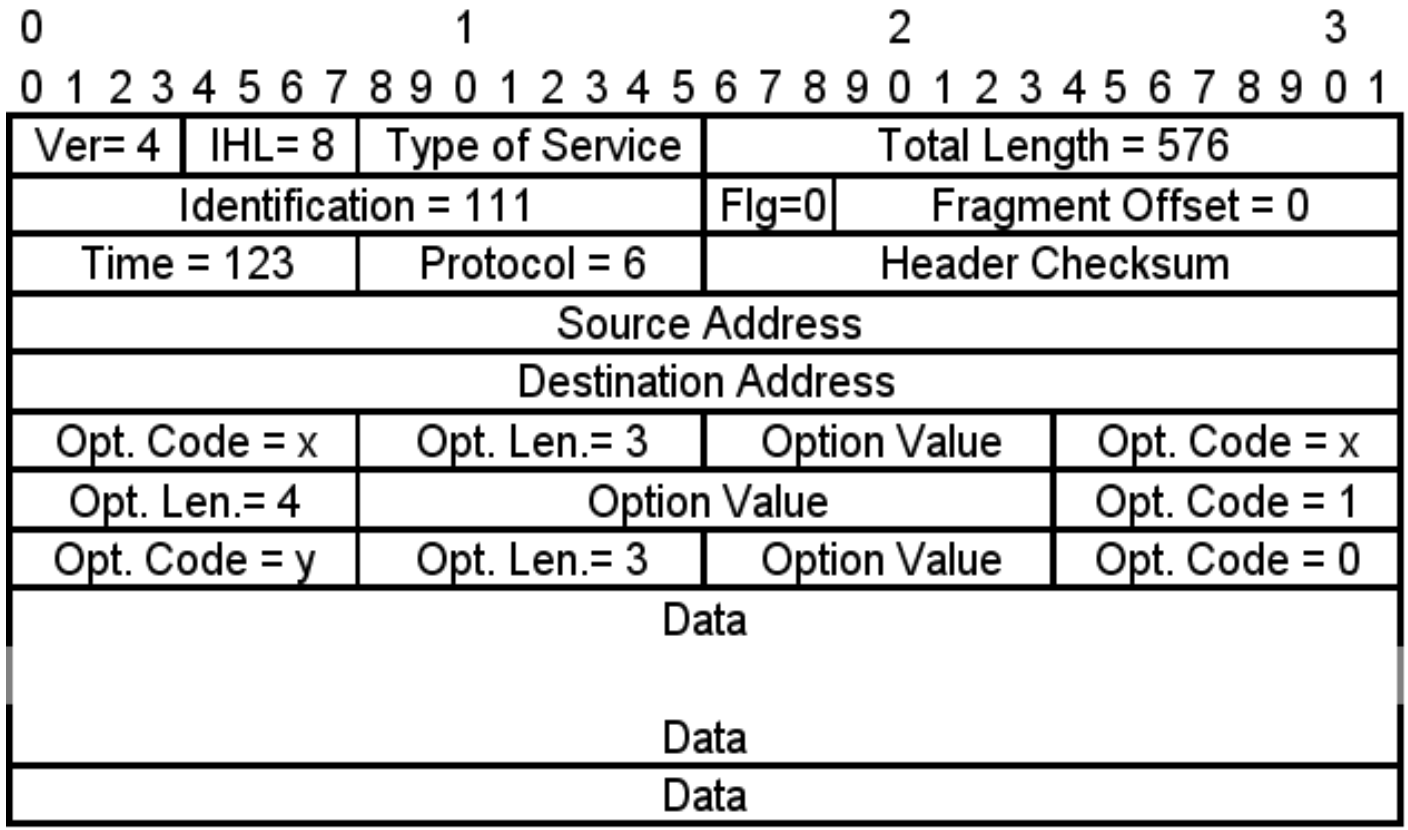
Przykład
2b: Pierwszy fragment



Przykład
2c: Drugi fragment



Przykład
3: Pakiet zawierający opcje



Kolejność transmisji danych

0	1	2	3
0 1 2 3 4 5 6 7	8 9 0 1 2 3 4 5	6 7 8 9 0 1 2 3	4 5 6 7 8 9 0 1
1	2	3	4
5	6	7	8
9	10	11	12

- Nagłówek i dane przesyłane oktetami
- Kolejność przesyłania taka jak łacińska kolejność czytania – przykład powyżej
- Znaczenie bitów – przykład poniżej

0	1	2	3	4	5	6	7
1	0	1	0	1	0	1	0

Adres IP

- **32-bitowy** adres identyfikujący węzeł (komputer) w Internecie
- Każdy węzeł to adres IP
- Zawartość adresu IP – identyfikatory:
  - **Sieci**
  - ***Hosta***
- Adres jest zwykle reprezentowany w notacji kropkowo-dziesiętnej
- Dziesiętne wartości każdego oktetu są odseparowane kropkami
- Przykład: **149.156.114.112**
- Adresy IP mogą być konfigurowane:
  - statycznie,
  - dynamicznie przez DHCP

Adresowanie i formaty adresów

Wyższe bity adresu	Format	Klasa
0	7 bitów na sieć, 24 bity na host	A
10	14 bitów na sieć, 16 bitów na host	B
110	21 bitów na sieć, 8 bitów na host	C
111	Furtka do rozszerzonego adresowania	

Konfiguracja urządzeń w sieci – pojęcia podstawowe (1/2)

- Adres IP
  - Adres komputera
  - Unikalny w danej rutowalnej sieci
  - Np.: **192.168.0.2**
- Maska podsieci
  - Zakres podsieci, określony bitowo
  - Np.: **255.255.255.0**
  - Często szesnastkowo: **FF.FF.FF.00**
  - Lub dwójkowo: **11111111.11111111.11111111.00000000**

Konfiguracja urządzeń w sieci – pojęcia podstawowe (2/2)

- Adres podsieci
  - Iloczyn logiczny adresu IP i maski podsieci
  - (Adres IP)&(Maska podsieci)
  - Np.: **192.168.0.0**
- Adres rutera
  - Okno na świat danej podsieci
  - Powinien zawierać się w podsieci
  - Np.: **192.168.0.1**

Konfiguracja urządzeń w sieci – adresy IP i klasy (1/2)

- Klasa A
  - Zakres: od **0.0.0.0** do **127.255.255.255**
  - Sieci klasy **A** jest mniej niż **128**, ale każda może się składać z **milionów** urządzeń sieciowych
- Klasa B
  - Zakres: od **128.0.0.0** do **191.255.255.255**
  - Są **tysiące** sieci klasy **B**, z których każda może zawierać **tysiące** adresów
- Klasa C
  - Zakres: od **192.0.0.0** do **223.255.255.255**
  - Są **miliony** sieci klasy **C**, ale każda może liczyć nie więcej niż **254** urządzenia sieciowe

Konfiguracja urządzeń w sieci – adresy IP i klasy (2/2)

- Klasa D
  - Zakres: od **224.0.0.0** do **239.255.255.255**
  - Zarezerwowane; tzw. adresy grupowe niepowiązane z żadną siecią
- Klasa E
  - Zakres: od **240.0.0.0** do **247.255.255.255**
  - Do celów eksperymentalnych
- Wyjątki…

Konfiguracja urządzeń w sieci
Adresy (klasy) specjalne

- Pewne adresy zostały **zarezerwowane** do korzystania wyłącznie w sieciach nie podłączonych bezpośrednio do Internetu (sieci lokalne). Adresy te nie są rutowalne.
  - **10.*.*.***
  - **172.16.*.***
  - **192.168.*.*.**
- Każdy komputer ma dodatkowo adres **127.0.0.1** – tj. tzw. ***loopback***, czyli pętla umożliwiająca aplikacjom wysyłanie i odbieranie pakietów TCP/IP w obrębie tej samej maszyny.

Konfiguracja urządzeń w sieci
Adresy sieci i broadcast (1)

- Dla każdej podsieci:
  - Adres początkowy: **A.B.C.D$_{min}$ = sieć**
  - Adres końcowy: **A.B.C.D$_{max}$ = rozgłoszeniowy**
- Przykład:
  - Podsieć: **192.168.0.0**
  - Maska (klasa C): **255.255.255.0**
  - Zakres: **192.168.0.[0-255]**
  - Adres sieci: **192.168.0.0**
  - Adres rozgłoszeniowy: **192.168.0.255**
  - Pierwszy wolny adres: **192.168.0.1** (ruter)
  - Ostatni wolny adres: **192.168.0.254**

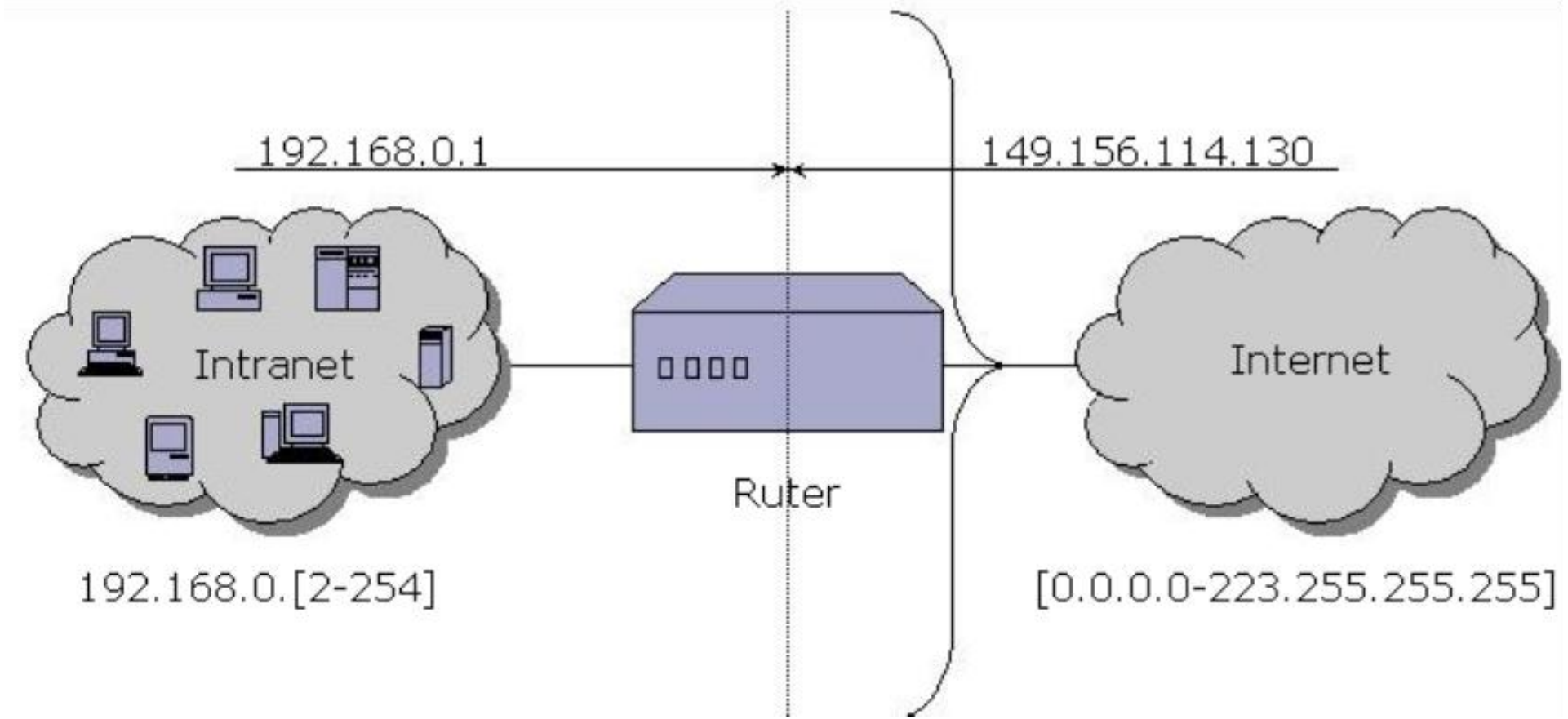
Konfiguracja urządzeń w sieci
Adresy sieci i broadcast (2)

- Wniosek 1: generalnie, liczba wolnych adresów równa jest $2^{liczba\_binarnych\_zer\_maski}-2$
- Wniosek 2: w typowej podsieci klasy C mamy 8 binarnych zer maski czyli 254 wolne adresy
- Wniosek 3: najmniejsza sensowna podsieć ma 2 binarne zera maski i liczy 4 adresy z czego 2 są wolne
- Wniosek 4: taka podsieć (4 adresy, 2 wolne) umożliwia spięcie tylko np.:
  - Dwóch komputerów
  - Komputera i rutera

Konfiguracja urządzeń w sieci
Powiększanie puli adresów (1)

- Konwersja adresów:
  - NAT (Network Address Translation)
  - „Maskarada” („Masquerading”)
- Możliwość „schowania” całych podsieci pod jednym adresem
- Obecnie bardzo popularne rozwiązanie

Konfiguracja urządzeń w sieci
Powiększanie puli adresów (2)



Konfiguracja urządzeń w sieci Ćwiczenia (1)

- Przypadek 1:
  - IP: **192.168.0.2**
  - Maska: **FF.FF.FF.00**
  - Sieć: **192.168.0.0**
  - Ruter: **172.16.0.1**
- Przypadek 2:
  - IP: **192.168.0.2**
  - Maska: **FF.FF.FF.00**
  - Sieć: **172.16.0.0**
  - Ruter: **192.168.0.1**
- Przypadek 3:
  - IP: **192.168.0.2**
  - Maska: **FF.FF.FF.80**
  - Sieć: **192.168.0.0**
  - Ruter: **192.168.0.129**
- Przypadek 4:
  - IP: **192.168.0.2**
  - Maska: **FF.00.00.00**
  - Sieć: **192.168.0.0**
  - Ruter: **192.168.0.1**

Konfiguracja urządzeń w sieci Ćwiczenia (2)

- Przypadek 5:
  - IP: **127.0.0.1**
  - Maska: **FF.FF.FF.00**
  - Sieć: **192.168.0.0**
  - Ruter: **192.168.0.1**
- Przypadek 6:
  - IP: **192.168.0.2**
  - Maska: **FF.FF.FF.00**
  - Sieć: **192.168.0.0**
  - Ruter: **192.168.0.1**

Protokół BGP-4
Wprowadzenie (1)

- Protokół wymiany informacji o rutingu pomiędzy niezależnymi sieciami
- Stworzony (w wersji pierwszej, jako “RFC: 1267”) w 1991 roku przez:
  - **Cisco Systems**
  - **IBM**
- Następca protokołu EGP (“RFC: 904”)
- Wymiana informacji o dostępności sieci z innymi systemami BGP
- Dostępność sieci to m.in. lista Systemów Autonomicznych (Autonomous Systems, APs), przez które przechodzi informacja o dostępności

Protokół BGP-4
Wprowadzenie (2)

- Ruting “hop-by-hop”
- Korzystanie z protokołów zapewniających pewność transmisji
- Nie potrzeba implementować:
  - **fragmentacji**
  - **retransmisji**
  - **potwierdzeń**
  - **sortowania**

- Konstrukcja grafu połączeń AS:
  - **eliminacja pętli rutingu**
  - **podejmowanie decyzji na poziomie rutingu AS**
- Nowy mechanizm wspierający bezklasowy ruting międzydomenowy
- Eliminacja klas sieci w rutingu BGP
- Agregacja ścieżek

Protokół BGP-4
Wprowadzenie (3)

- Oprócz wbudowanego mechanizmu uwierzytelniania, możliwość użycia w BGP, dowolnego mechanizmu uwierzytelniania protokołu transportowego
- Założenie, że protokół transportowy nie zrywa połączeń
- Użycie TCP jako protokołu transportowego, jako:
  - **spełniającego wymagania BGP**
  - **dostępnego w praktycznie wszystkich komrecyjnie dostępnych ruterach i komputerach**
- Użycie portu TCP o numerze 179 do ustanawiania połączeń

Literatura

- Defense Advance Research Project Agency, “RFC (Request For Comments): 791, Internet Protocol”, http://www.ietf.org/rfc/rfc0791.txt
- Network Working Group, “RFC (Request For Comments): 1771, A Border Gateway Protocol 4 (BGP-4)”, http://www.ietf.org/rfc/rfc1771.txt
- “NETWORLD – Wprowadzenie do sieci”, http://www.networld.pl/artykuly/20226.html
- Cerf, V., "The Catenet Model for Internetworking," Information Processing Techniques Office, Defense Advanced Research Projects Agency, IEN 48, July 1978
- Bolt Beranek and Newman, "Specification for the Interconnection of a Host and an IMP," BBN Report
- Wikipedia, http://www.wikipedia.pl/

Literatura.

- **"T.J.K. – Program Edukacyjny o Sieciach Telekomunikacyjnych – Pomoc"**
- **"Topologie równoległe"** [http://icis.pcz.czest.pl/materials/topologie/]
- **"Interaktywny podręcznik sieci komputerowych"** [http://www.man.poznan.pl/~pawelw/dyplom/]
- **"Topologie sieci"** [http://www.republika.pl/legecki/topologie.html]
- **"Grafy"** [http://student.uci.agh.edu.pl/~wasikows/grafy/]
- **Strona główna – wikipedia** [http://pl.wikipedia.org/wiki/Strona_g%C5%82%C3%B3wna]