Routing

część 2: tworzenie tablic

Sieci komputerowe

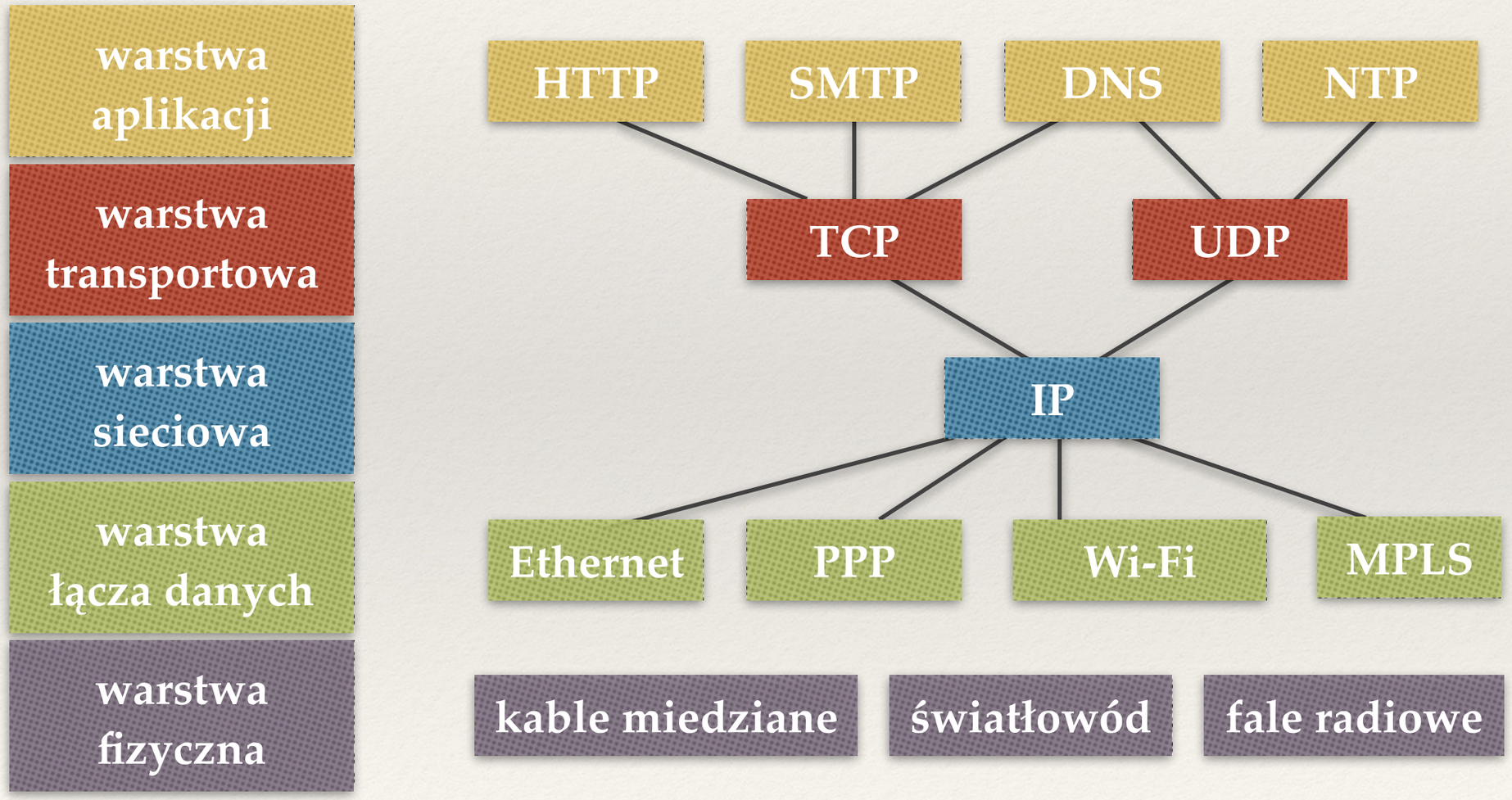
Wykład 3

Marcin Bieńkowski

W poprzednim odcinku

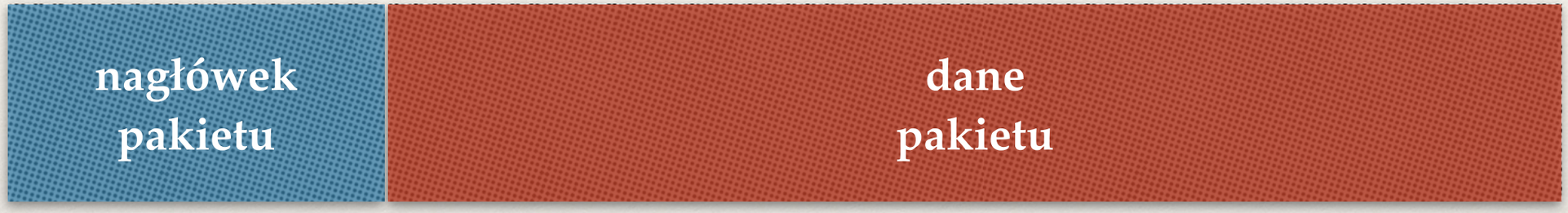
Jedna warstwa sieci i globalne adresowanie

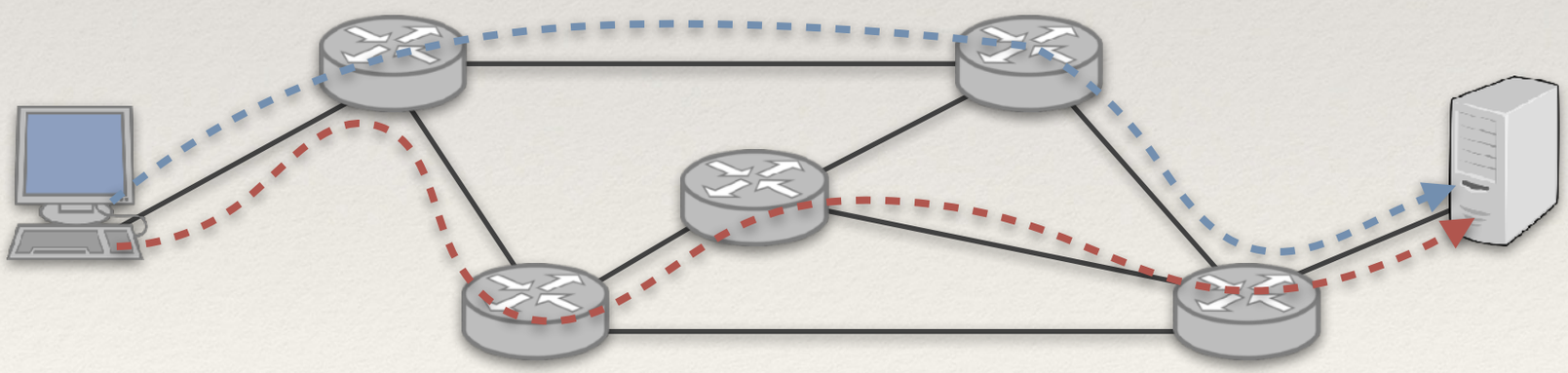
- Każde urządzenie w sieci posługuje się tym samym protokołem warstwy sieci. W Internecie: protokół IP.
- Każde urządzenie ma unikatowy adres. W Internecie: adresy IP.



Przełączanie pakietów

- Chcemy przesyłać między aplikacjami strumień danych.
- Wysyłany strumień danych dzielimy na małe porcje: pakiety.
- Każdy pakiet przesyłany niezależnie.





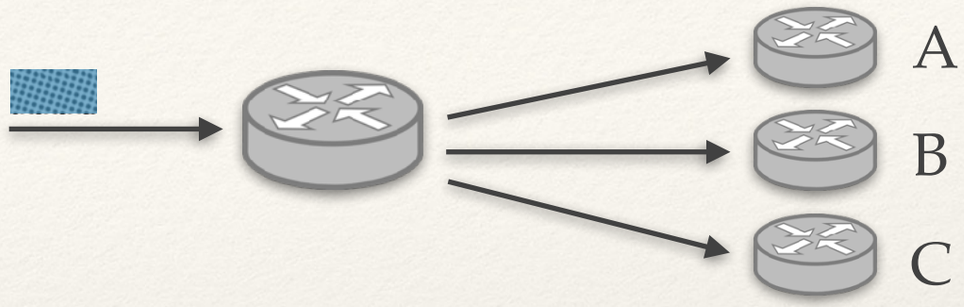
Notacja CIDR

- CIDR opisuje prefiksy adresów IP:
  - `156.17.4.32` = `10011100.00010001.00000100.00100000`
  - `156.17.4.32/28` = adresy zaczynające się od prefiksu 28-bitowego `10011100.00010001.00000100.0010`

- Zazwyczaj sieć może być opisana jednym prefiksem CIDR.

Tablice routingu

prefiks CIDR	akcja
`0.0.0.0/0`	do routera A
`156.17.4.0/24`	do routera B
`156.17.4.128/25`	do routera C
`156.17.4.128/26`	do routera B



- Router podejmuje decyzję na podstawie nagłówka pakietu w oparciu o tablice routingu.
- Zawiera reguły typu „jeśli adres docelowy pakietu zaczyna się od prefiksu *A*, to wyślij pakiet do *X*“.
- Pakiet niepasujący do żadnej reguły jest odrzucany.

Dziś: tworzenie tablic

Ręczna konfiguracja routingu

- Sprawdza się w przypadku małej sieci.
- W Internecie bez szans powodzenia:
  - dodawane lub usuwane routery i łącza,
  - zmiany parametrów i awarie łączy.
- Chcemy zapewniać łączność i unikać cykli w routingu (pakietów krążących w kółko)

Tablica przekazywania i routingu

- **Tablica przekazywania (*forwarding table*).**
  - Do tej pory nazywaliśmy ją tablicą routingu.
  - Informacje **o następnym routerze na trasie**.

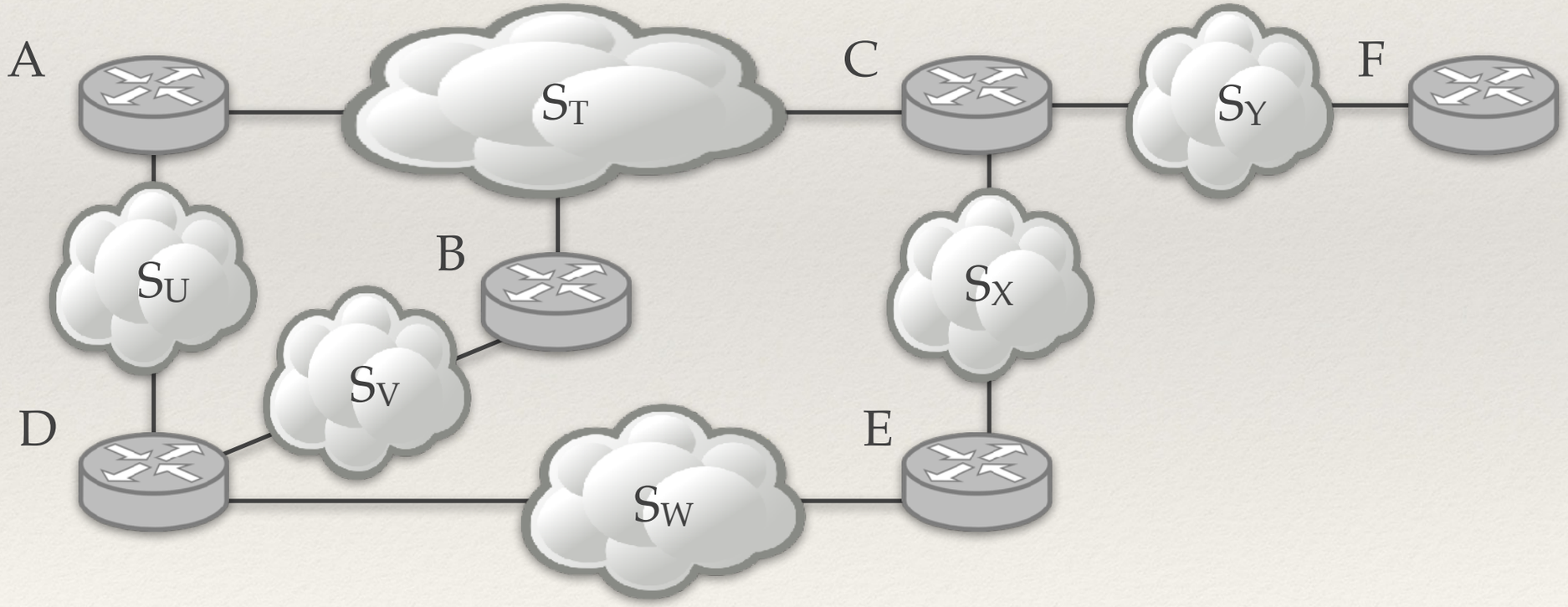
prefiks CIDR	akcja
`156.17.4.0/24`	do routera B
`156.17.4.128/25`	do routera C
`156.17.4.128/26`	do routera B

  - Do podejmowania decyzji o pakietach na podstawie najdłuższego pasującego prefiksu.
  - Zoptymalizowana struktura danych wspomagana sprzętowo.

- **Tablica routingu (*routing table*).**
  - Informacje o **trasach**.
  - Zawiera dodatkowe informacje, np. zapasowe trasy routingu.

Cel: konfiguracja tablic przekazywania

- Do dowolnej sieci.
- Bez tworzenia cykli w routingu.
- Implementacja w rozproszonym środowisku.

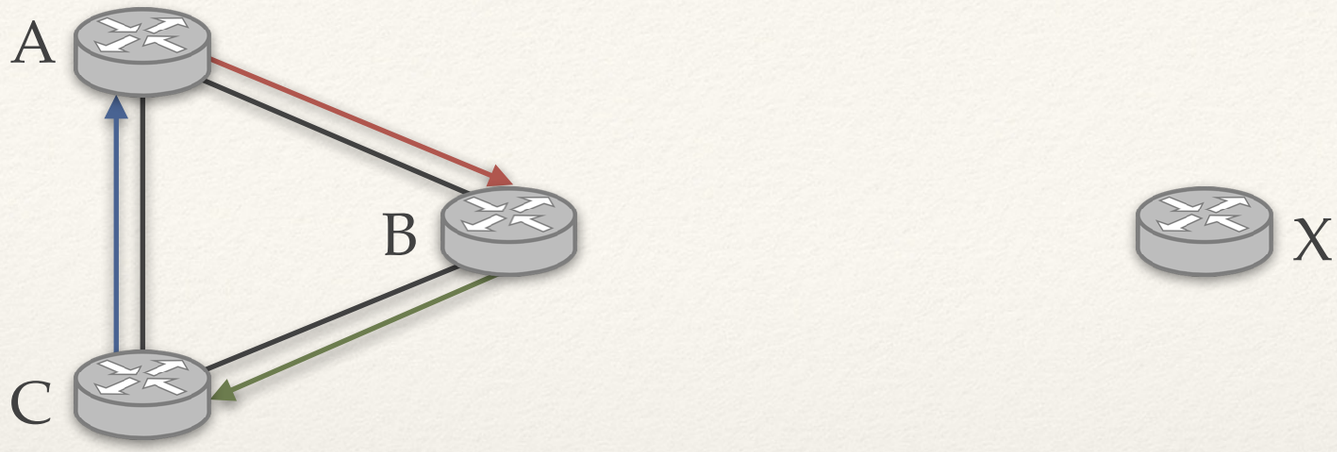


Najkrótsze ścieżki

- Chcemy dodatkowo wybierać trasy minimalizujące „odległość do celu” (sumę wag na krawędziach na trasie do celu).

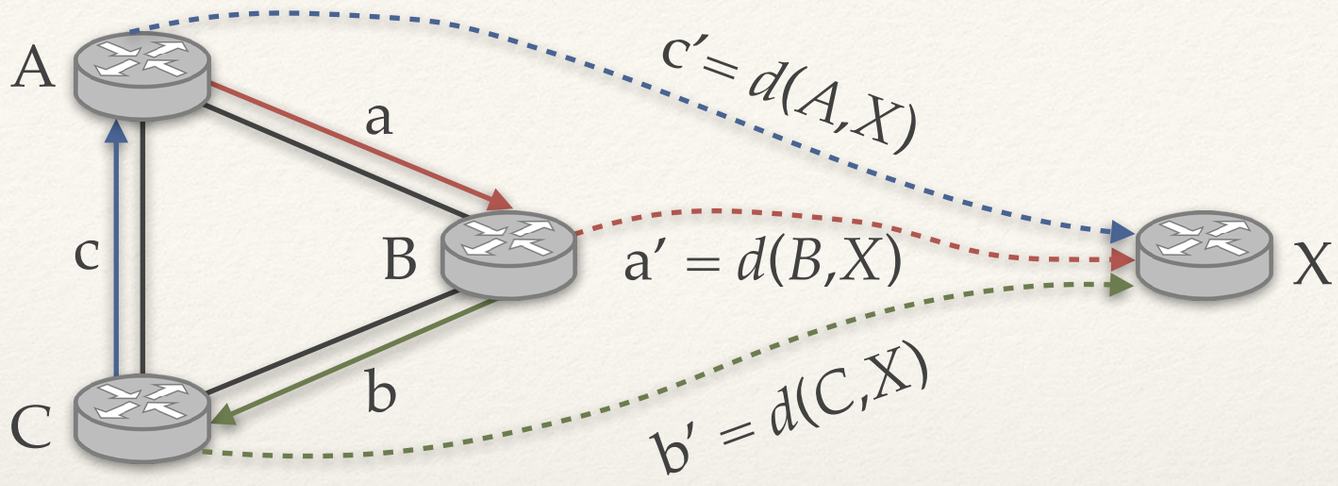
- Jak zdefiniować wartości krawędzi? (**metryka**):
  - czas propagacji;
  - koszt pieniężny;
  - wszędzie 1
    (wtedy odległość = 1 + liczba routerów na trasie).

Routing według najkrótszych ścieżek → brak cykli



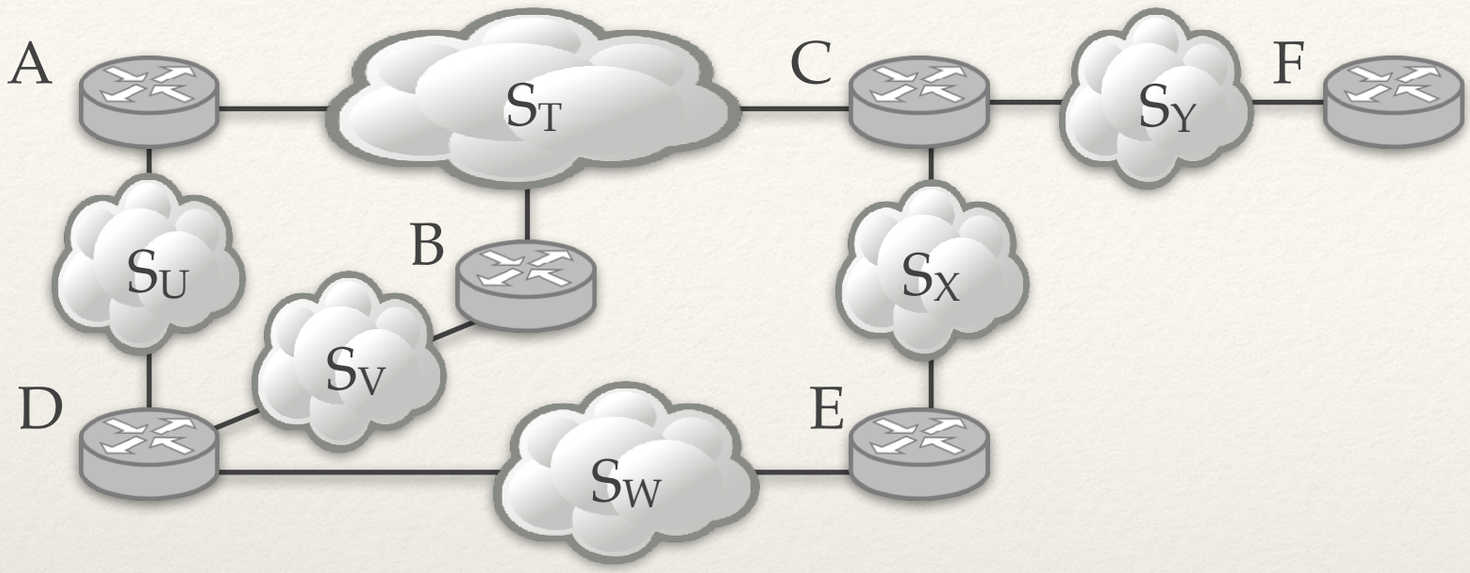
- Trasy do X (obliczone na poszczególnych routerach).
- Załóżmy że mamy cykl ($A \to B \to C \to A$).

Routing według najkrótszych ścieżek → brak cykli



- Trasy do X (obliczone na poszczególnych routerach).
- Załóżmy że mamy cykl ($A \to B \to C \to A$).
- Ponieważ wybrane ścieżki są najkrótsze, to:
  - $a + a' \leq c'$
  - $b + b' \leq a'$
  - $c + c' \leq b'$
- A zatem: $a + b + c \leq 0$ → sprzeczność.

Bezpośrednio podłączone sieci

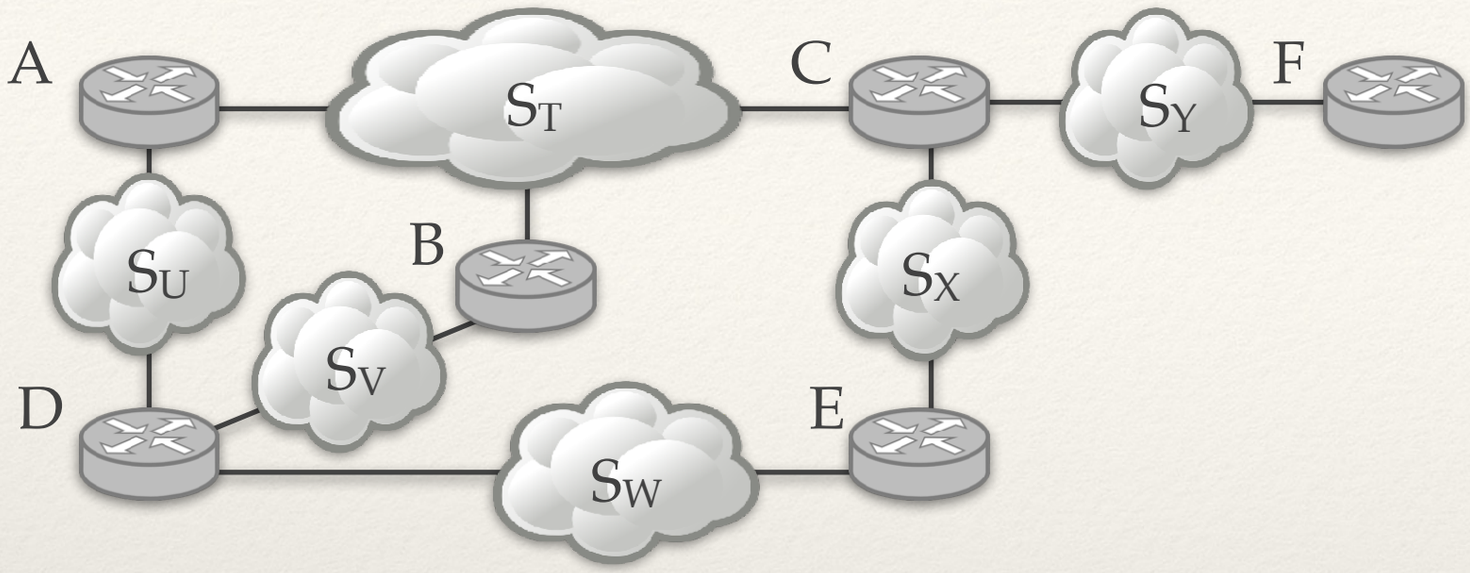


sąsiedztwo routera D	
sieć	**odległość**
S_U	1
S_V	1
S_W	1

Założenia:

- Każdy router zna sieci z którymi jest połączony bezpośrednio.
- Router zna stan sąsiadujących łączy przez okresowy monitoring, np. wymiana pakietów co 30 sekund z sąsiadem.

Bezpośrednio podłączone sieci



sąsiedztwo routera D	
sieć	**odległość**
S_U	1
S_V	1
S_W	1

To jest również odległość do dowolnego routera mającego kartę sieciową w sieci S_W (np. do E).

Założenia:

- Każdy router zna sieci z którymi jest połączony bezpośrednio.
- Router zna stan sąsiadujących łączy przez okresowy monitoring, np. wymiana pakietów co 30 sekund z sąsiadem.

Dwa typy algorytmów

- **Algorytmy stanu łączy**
  - Powiadom wszystkich o sieciach, do których jesteś połączony bezpośrednio.
  - Na podstawie takich sąsiedztw zbuduj graf sieci i oblicz lokalnie najkrótsze ścieżki.

- **Algorytmy wektora odległości**
  - Okresowo powiadamiaj sąsiednie routery o całej swojej tablicy przekazywania.
  - Aktualizuj swoją tablicę routingu na tej podstawie.

Stan łączy

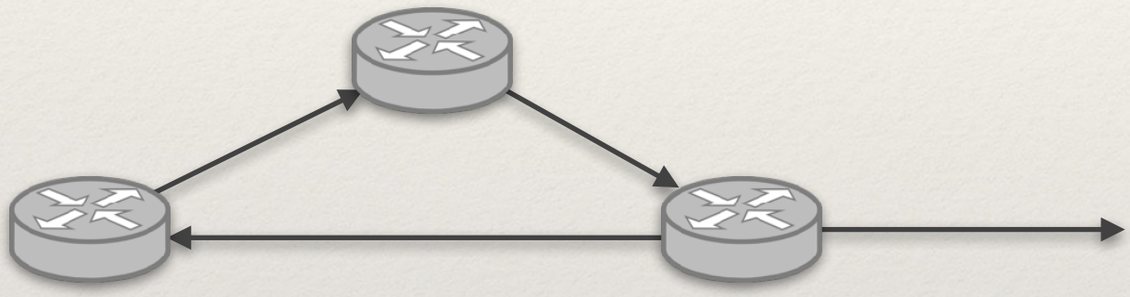
Algorytm stanu łączy

- Wysyłanie informacji o sąsiedztwie do wszystkich routerów.
  - **Jak wysłać wiadomość do wszystkich w sieci skoro nie mamy jeszcze tablic routingu?**

- Lokalne obliczenie najkrótszych ścieżek (od jednego źródła).
  - Pamięć = graf o wielkości $O(|V|+|E|)$.
  - Czas $O(|V| \log |V| + |E|)$, algorytm Dijkstry.

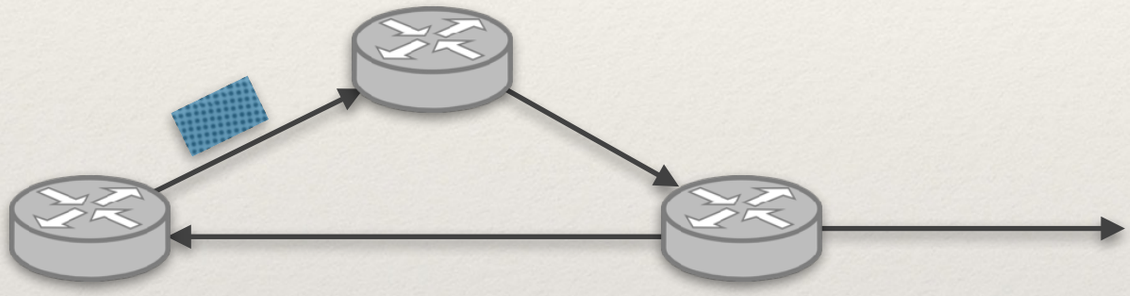
Niekontrolowane „zalewanie” sieci informacją

- Reguła: po odebraniu informacji *q* od routera *X*, wyślij *q* do wszystkich sąsiadów poza *X*.
- Problem:



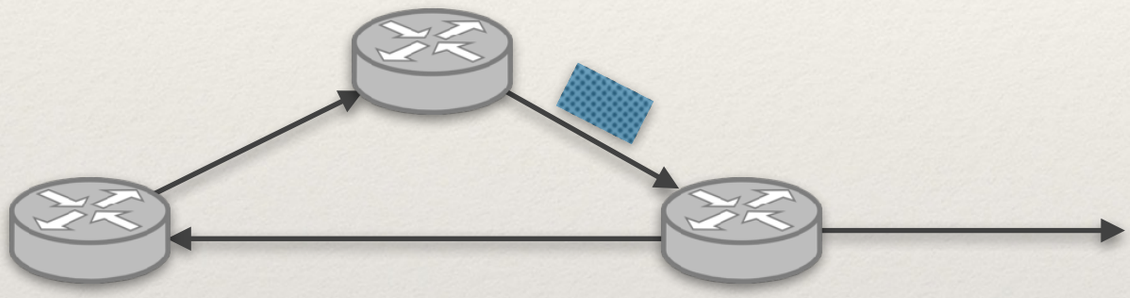
Niekontrolowane „zalewanie” sieci informacją

- Reguła: po odebraniu informacji *q* od routera *X*, wyślij *q* do wszystkich sąsiadów poza *X*.
- Problem:



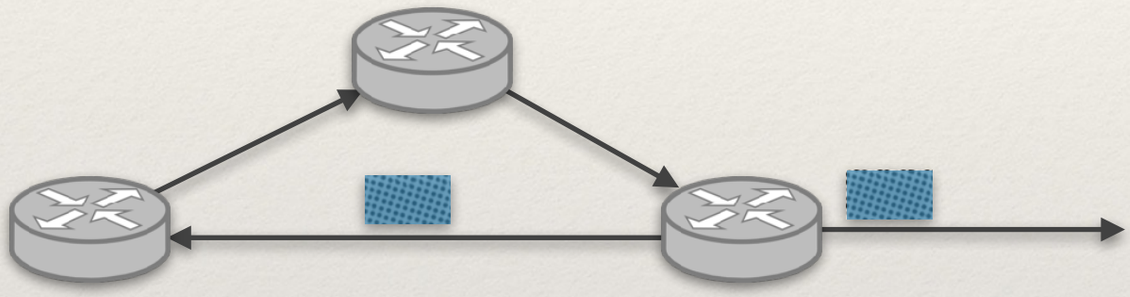
Niekontrolowane „zalewanie” sieci informacją

- Reguła: po odebraniu informacji *q* od routera *X*, wyślij *q* do wszystkich sąsiadów poza *X*.
- Problem:



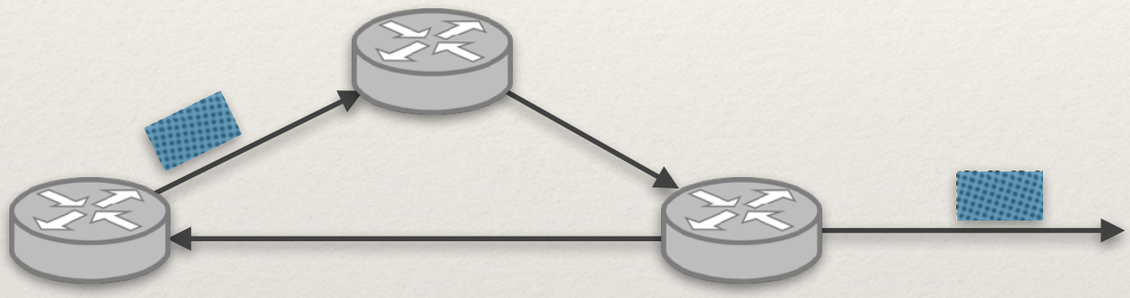
Niekontrolowane „zalewanie” sieci informacją

- Reguła: po odebraniu informacji *q* od routera *X*, wyślij *q* do wszystkich sąsiadów poza *X*.
- Problem:



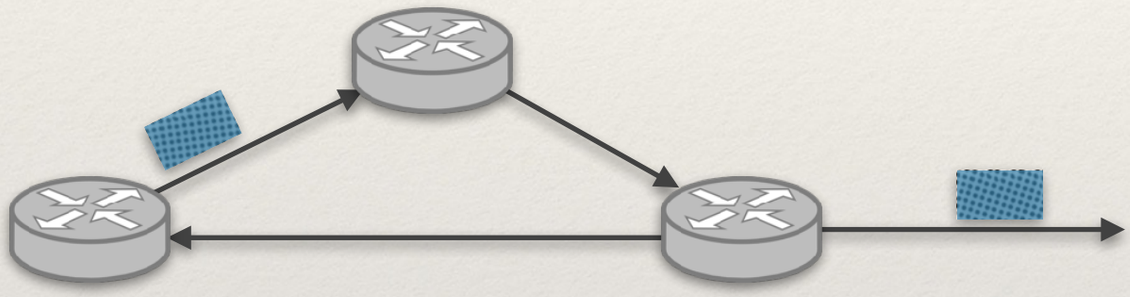
Niekontrolowane „zalewanie” sieci informacją

- Reguła: po odebraniu informacji *q* od routera *X*, wyślij *q* do wszystkich sąsiadów poza *X*.
- Problem:



Niekontrolowane „zalewanie” sieci informacją

- Reguła: po odebraniu informacji *q* od routera *X*, wyślij *q* do wszystkich sąsiadów poza *X*.
- Problem:



- Nawet jeśli w grafie nie ma cykli: wiele kopii pakietu może dotrzeć do jednego routera i każda z nich zostanie przesłana dalej.
- Trzeba pamiętać, jakie informacje już rozsyłaliśmy.

Kontrolowane „zalewanie” sieci informacją

- **Router źródłowy dodaje do informacji *q*:**
  - swój adres *s*,
  - numer sekwencyjny *n*.

- **Po odebraniu informacji *(q, s, n)* od routera *X*:**
  - Jeśli nie przekazywaliśmy jeszcze informacji z adresu *s* o numerze *n*, to wysyłamy *(q, s, n)* do wszystkich sąsiadów (poza *X*).
  - Trzymamy pary *(s, n)* przez pewien czas.

Zbieżność do stanu stabilnego

- Jeśli sieć nie zmienia się przez pewien czas, to:
  - każdy router będzie miał ten sam obraz sieci;
  - stworzone tablice przekazywania będą bez cykli w routingu.

- Możliwe cykle, jeśli niektóre routery już wiedzą o awarii łącza a inne nie (ćwiczenie).

Protokół OSPF (Open Shortest Path First).

- Implementacja algorytmu stanu łączy.
- Komunikaty LSA = *Link State Advertisement* (opisują stan pojedynczego łącza).
  - Przesyłane na początku, przy zmianie i co jakiś czas (~30 min.).
  - LSA zawiera źródło i numer sekwencyjny.
  - Przechowywane przez ~1h w pamięci.

Wektory odległości

Co robi router

- **Przechowuje wektor odległości *V***
  - V zawiera odległości do znanych mu routerów i sieci.
  - Początkowo: *V* = tylko sieci dostępne bezpośrednio.
- **Co pewien czas:**
  - Wysyła *V* do sąsiednich routerów.
  - Aktualizuje tablicę routingu na podstawie informacji otrzymanych od sąsiadów.
  - Tablica routingu = tablica przekazywania + informacja z *V* o odległościach do celu.

Aktualizacja tablicy dla routera X

A mówi: „mam do S_B odległość d(A,B)“.

$$d(X, S_B) \leftarrow \min \{\, d(X, S_B),\ \ d(X, S_A) + d(A, S_B) \,\}$$

Aktualna odległość od X do S_B.

A leży w sieci S_A połączonej bezpośrednio z X.

Aktualizacja tablicy dla routera X

A mówi: „mam do S_B odległość d(A,B)“.

$$d(X, S_B) \leftarrow \min \{\, d(X, S_B),\ \ d(X, S_A) + d(A, S_B) \,\}$$

Aktualna odległość od X do S_B.

A leży w sieci S_A połączonej bezpośrednio z X.

- Przy aktualizacji $d(X, S_B)$ ustawiamy A jako pierwszy router na trasie do S_B.
- Jeśli X nie zna S_B, to przyjmujemy $d(X, S_B) = \infty$.
- Rozproszony wariant algorytmu Bellmana-Forda.
- Przechowujemy tylko jedną (najlepszą) ścieżkę.

Aktualizacja tablicy dla routera X

A mówi: „mam do S_B odległość d(A,B)“.

$$d(X, S_B) \leftarrow \min \{ d(X, S_B),\ d(X, S_A) + d(A, S_B) \}$$

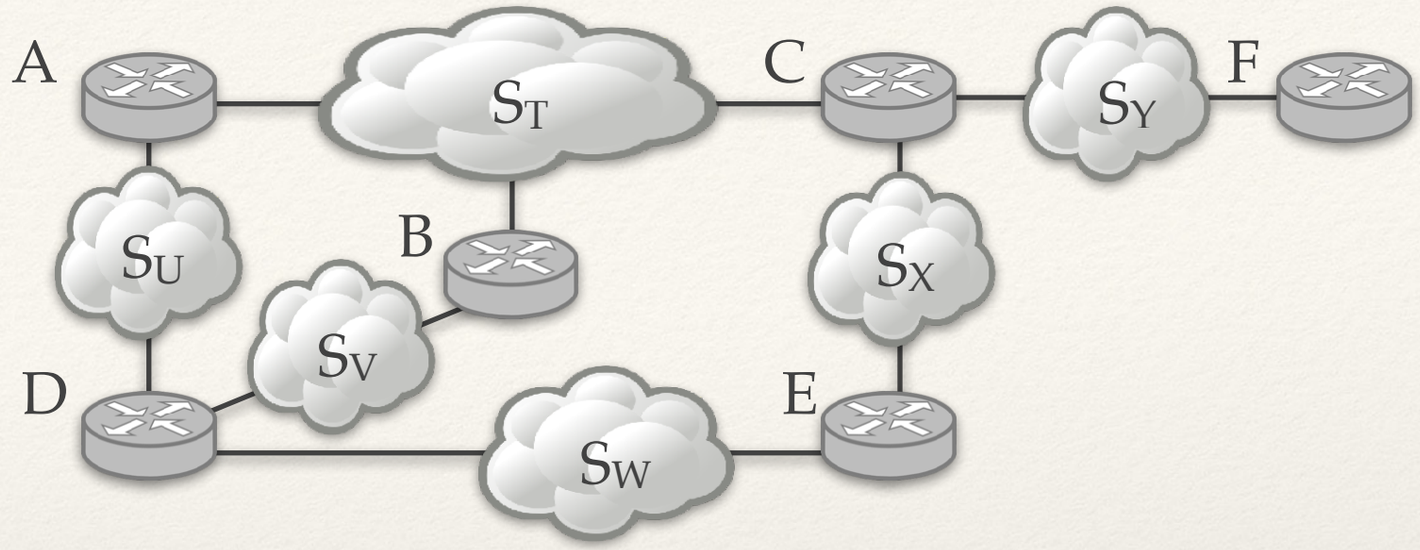
Aktualna odległość od X do S_B.

A leży w sieci S_A połączonej bezpośrednio z X.

- Przy aktualizacji $d(X, S_B)$ ustawiamy A jako pierwszy router na trasie do S_B.
- Jeśli X nie zna S_B, to przyjmujemy $d(X, S_B) = \infty$.
- Rozproszony wariant algorytmu Bellmana-Forda.
- Przechowujemy tylko jedną (najlepszą) ścieżkę.

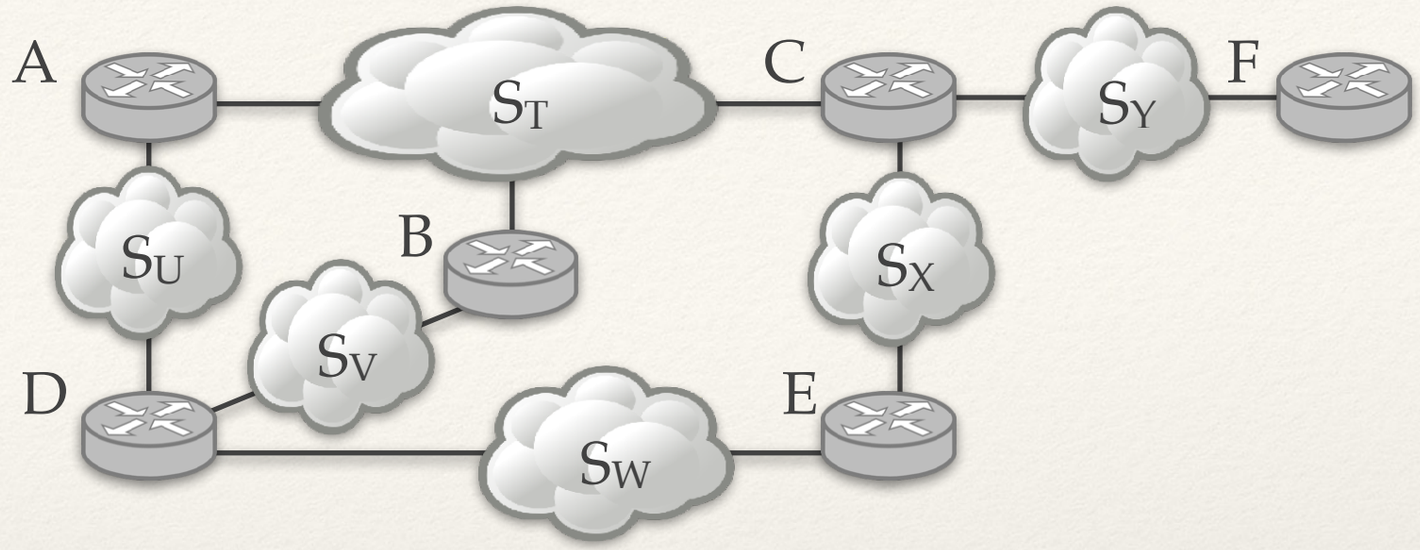
Teoretyczne założenie, w praktycznych implementacjach wiele.

Przykład tworzenia tablic: krok 0



	A	B	C	D	E	F
trasa do S_T	1	1	1			
trasa do S_U	1			1		
trasa do S_V		1		1		
trasa do S_W				1	1	
trasa do S_X			1		1	
trasa do S_Y			1			1

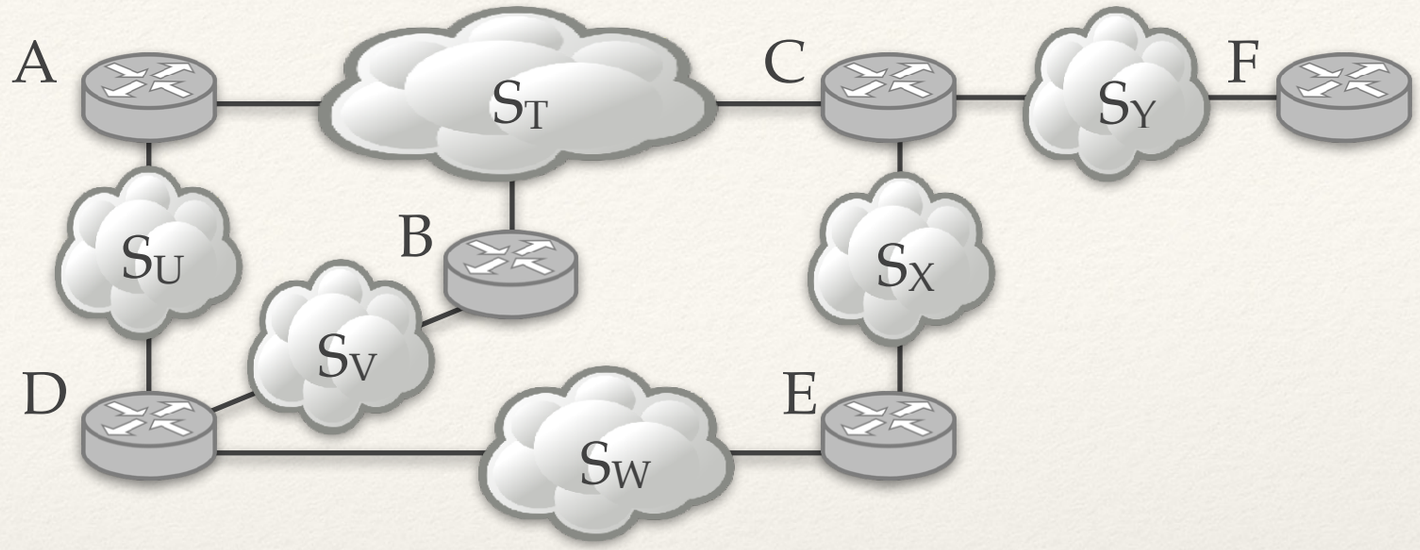
Przykład tworzenia tablic: krok 0



C jest sąsiadem A (w sieci S_T) odległym o 1.

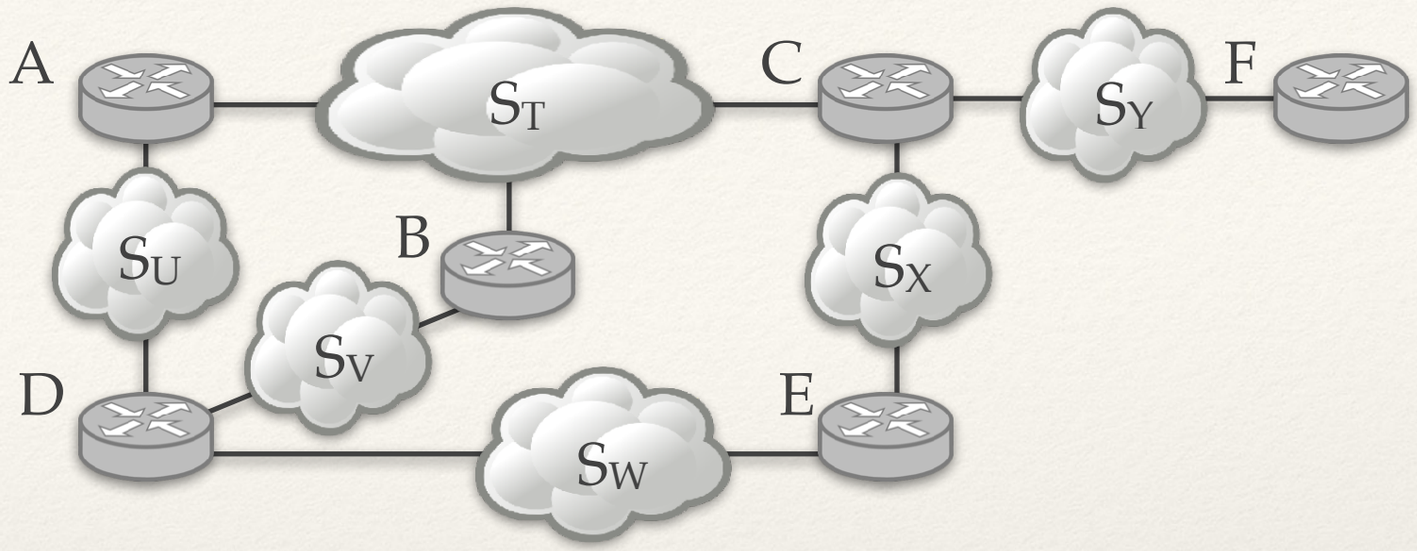
	A	B	C	D	E	F
trasa do S_T	1	1	1			
trasa do S_U	1			1		
trasa do S_V		1		1		
trasa do S_W				1	1	
trasa do S_X			1		1	
trasa do S_Y			1			1

Przykład tworzenia tablic: krok 1



	A	B	C	D	E	F
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)
trasa do S_Y	2 (via C)	2 (via C)	1		2 (via C)	1

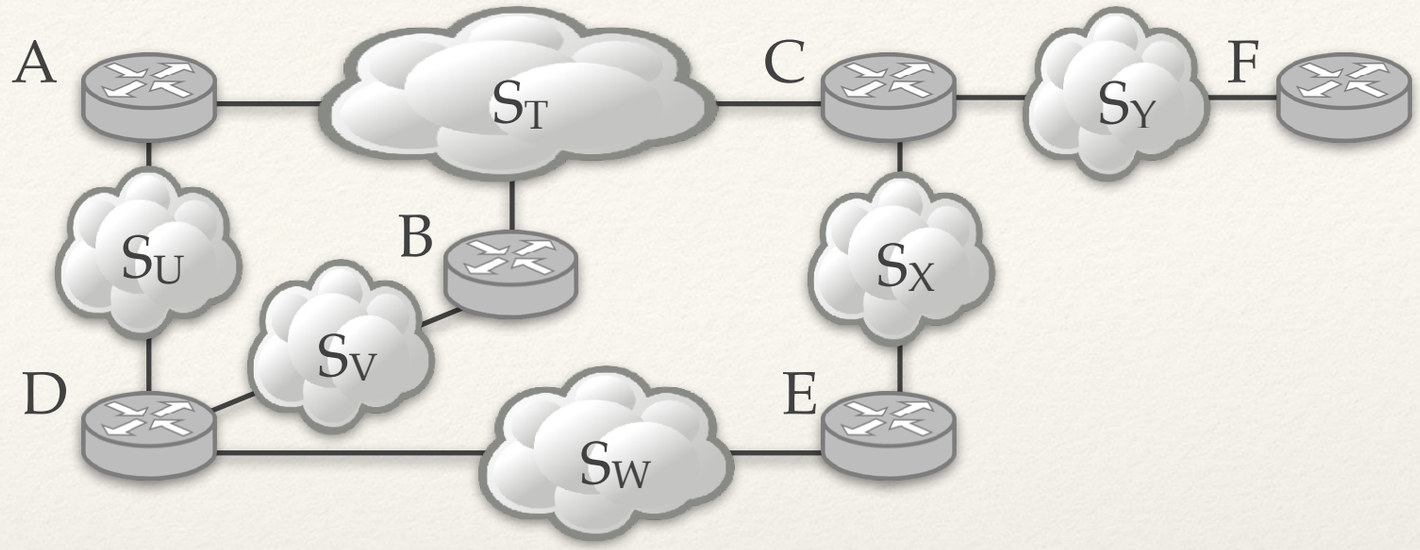
Przykład tworzenia tablic: krok 1



A dowiedział się o sieci S_V od B i od D, ale D powiadomił go szybciej.

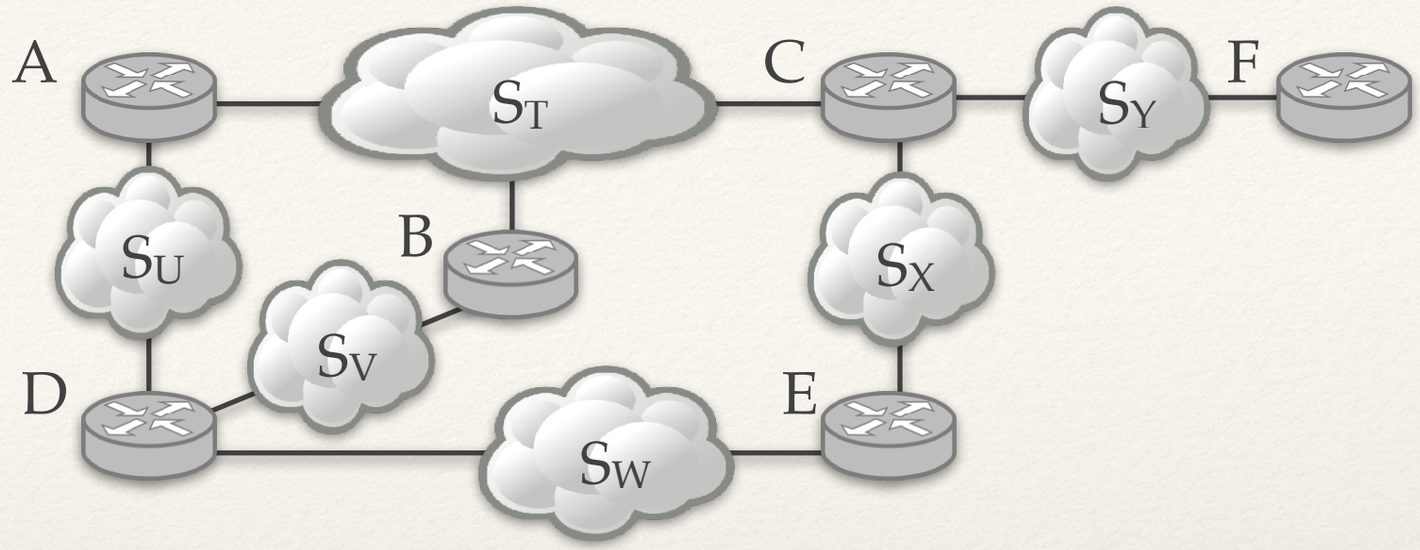
	A	B	C	D	E	F
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)
trasa do S_Y	2 (via C)	2 (via C)	1		2 (via C)	1

Przykład tworzenia tablic: krok 1



	A	B	C	D	E	F
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)
trasa do S_Y	2 (via C)	2 (via C)	1		2 (via C)	1

Przykład tworzenia tablic: krok 1

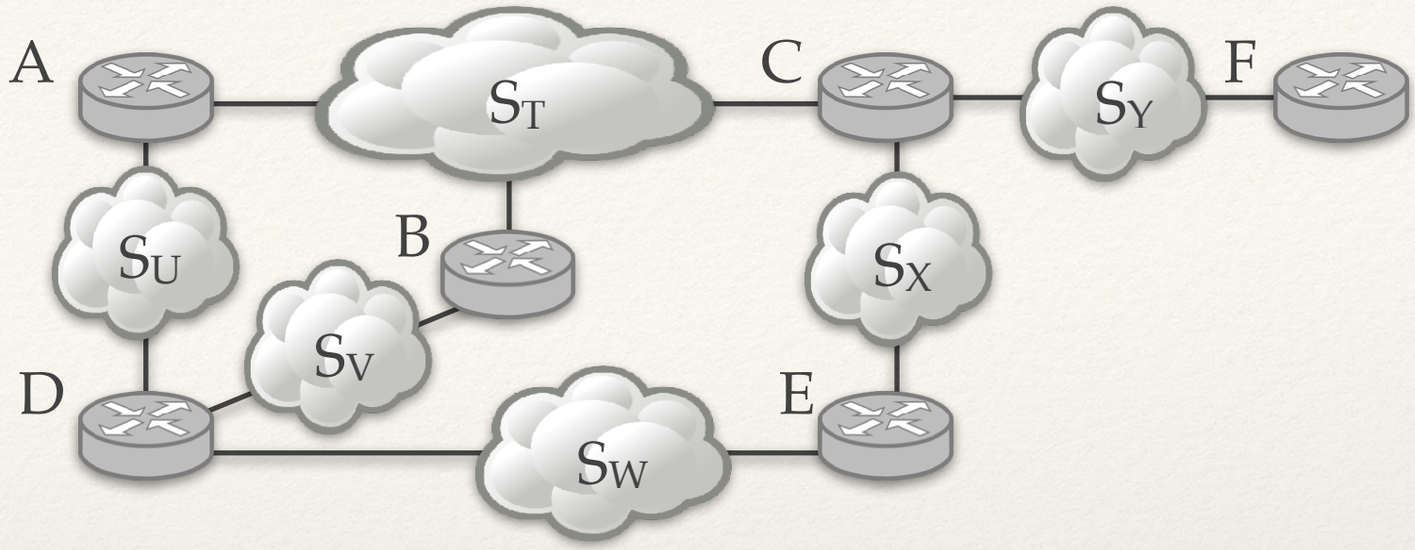


A jest sąsiadem D
(w sieci S_U) odległym o 1.

	A	B	C	D	E	F
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)
trasa do S_Y	2 (via C)	2 (via C)	1		2 (via C)	1

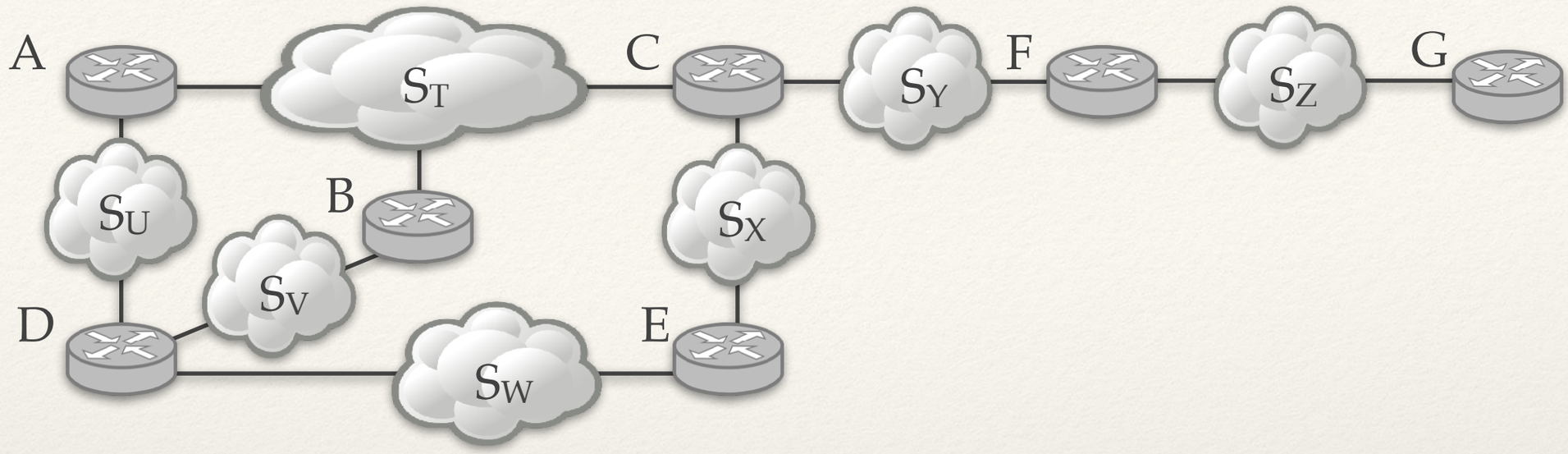
Przykład tworzenia tablic: krok 2

Stan stabilny = kolejne transmisje wektorów nie powodują aktualizacji.



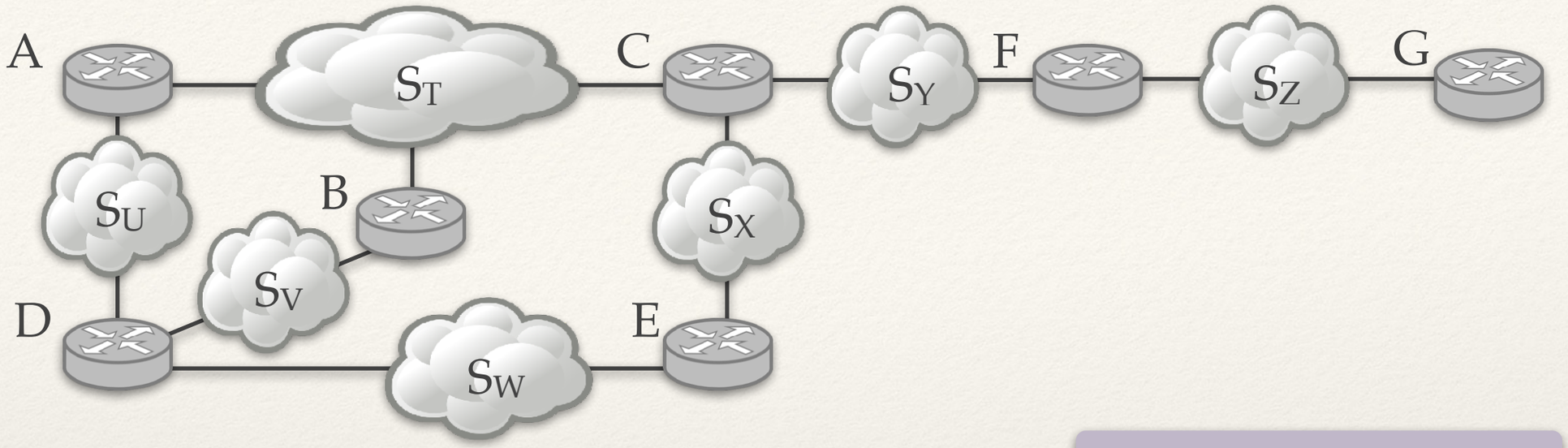
	A	B	C	D	E	F
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	3 (via C)
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	3 (via C)
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	3 (via C)
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)
trasa do S_Y	2 (via C)	2 (via C)	1	3 (via A)	2 (via C)	1

Dodawanie routera: krok 0



	A	B	C	D	E	F	G
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)	
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	3 (via C)	
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	3 (via C)	
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	3 (via C)	
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)	
trasa do S_Y	2 (via C)	2 (via C)	1	3 (via A)	2 (via C)	1	
trasa do S_Z						**1**	**1**

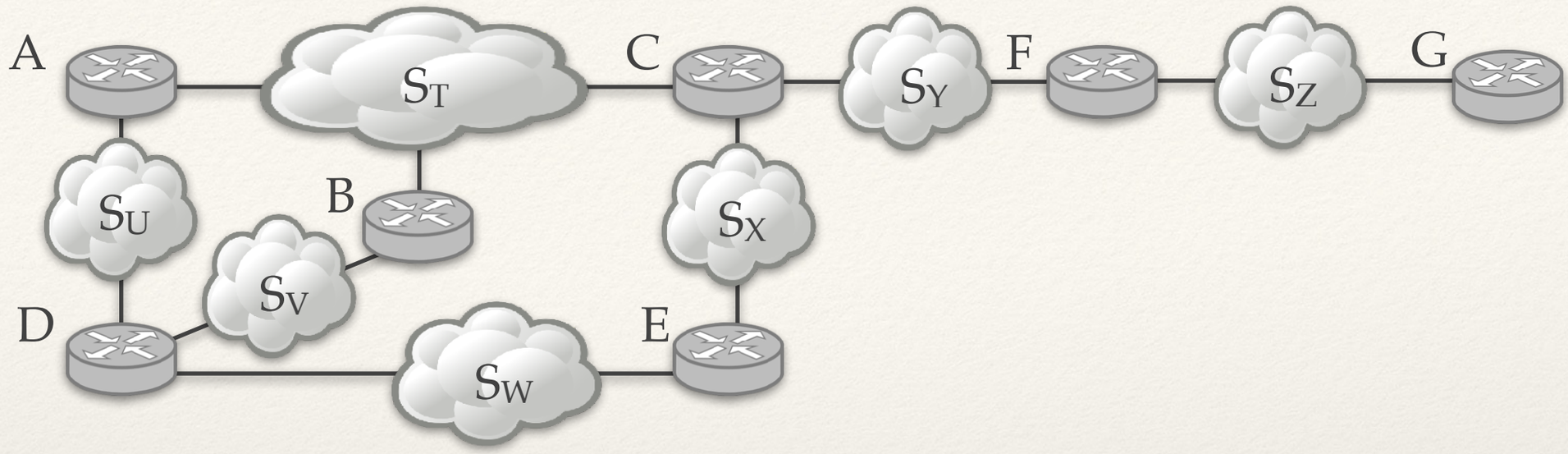
Dodawanie routera: krok 0



Dodane ręcznie

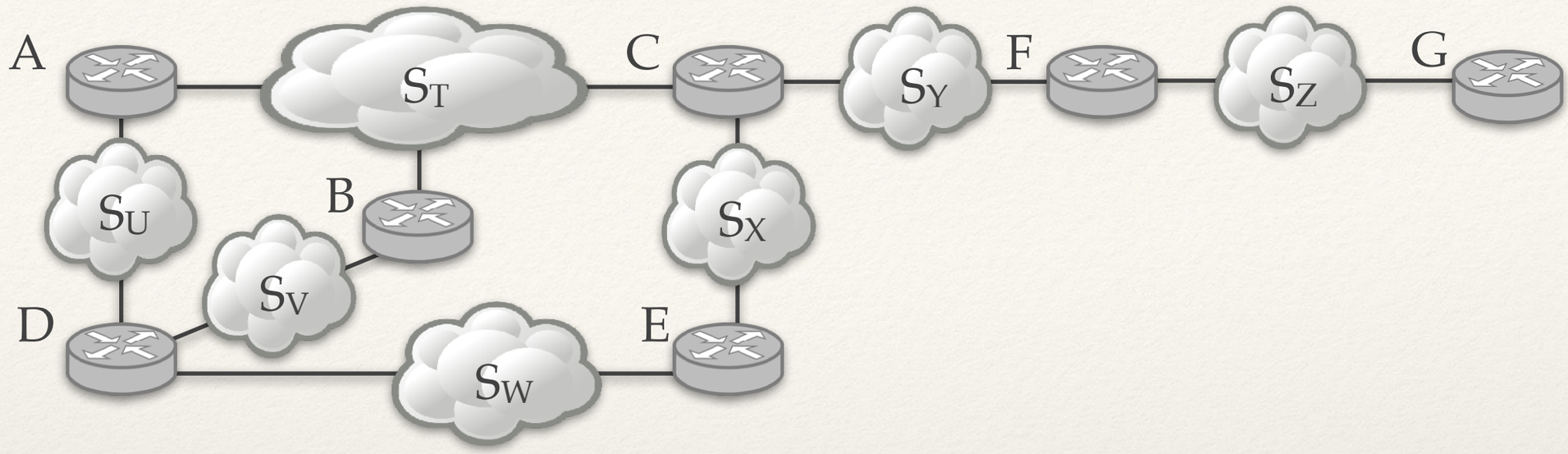
	A	B	C	D	E	F	G
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)	
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	3 (via C)	
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	3 (via C)	
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	3 (via C)	
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)	
trasa do S_Y	2 (via C)	2 (via C)	1	3 (via A)	2 (via C)	1	
trasa do S_Z						**1**	**1**

Dodawanie routera: krok 1



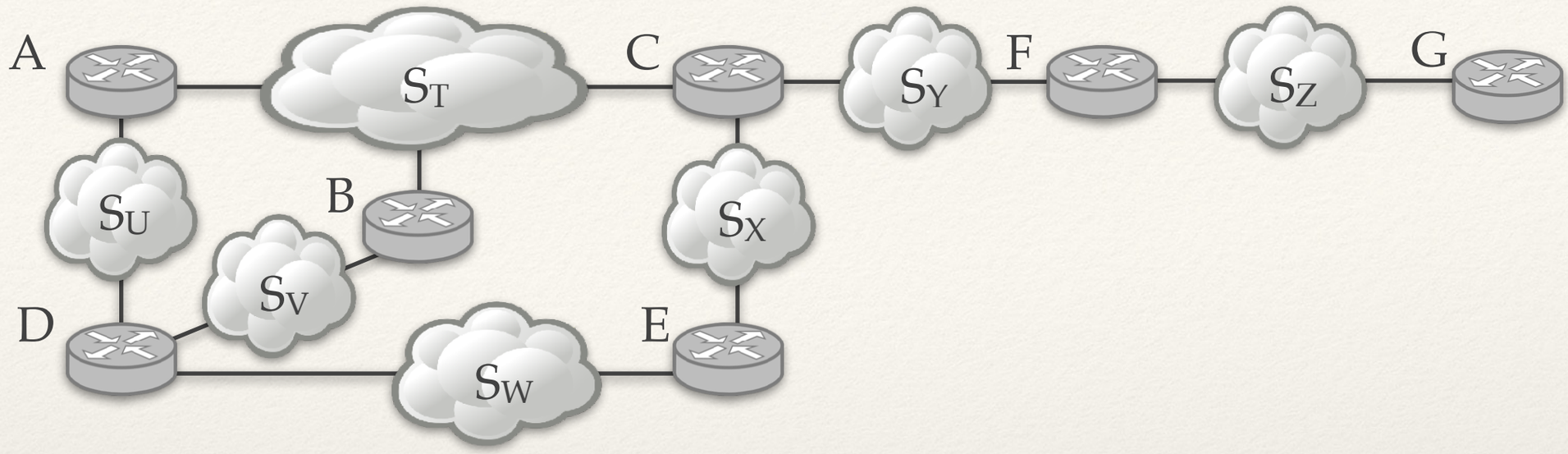
	A	B	C	D	E	F	G
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)	3 (via F)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	3 (via C)	4 (via F)
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	3 (via C)	4 (via F)
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	3 (via C)	4 (via F)
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)	3 (via F)
trasa do S_Y	2 (via C)	2 (via C)	1	3 (via A)	2 (via C)	1	2 (via F)
trasa do S_Z			2 (via F)			1	1

Dodawanie routera: krok 2



	A	B	C	D	E	F	G
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)	3 (via F)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	3 (via C)	4 (via F)
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	3 (via C)	4 (via F)
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	3 (via C)	4 (via F)
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)	3 (via F)
trasa do S_Y	2 (via C)	2 (via C)	1	3 (via A)	2 (via C)	1	2 (via F)
trasa do S_Z	3 (via C)	3 (via C)	2 (via F)		3 (via C)	1	1

Dodawanie routera: krok 3 (stan stabilny)



	A	B	C	D	E	F	G
trasa do S_T	1	1	1	2 (via B)	2 (via C)	2 (via C)	3 (via F)
trasa do S_U	1	2 (via A)	2 (via A)	1	2 (via D)	3 (via C)	4 (via F)
trasa do S_V	2 (via D)	1	2 (via B)	1	2 (via D)	3 (via C)	4 (via F)
trasa do S_W	2 (via D)	2 (via D)	2 (via E)	1	1	3 (via C)	4 (via F)
trasa do S_X	2 (via C)	2 (via C)	1	2 (via E)	1	2 (via C)	3 (via F)
trasa do S_Y	2 (via C)	2 (via C)	1	3 (via A)	2 (via C)	1	2 (via F)
trasa do S_Z	3 (via C)	3 (via C)	2 (via F)	4 (via B)	3 (via C)	1	1

Szybkość zbieżności

- Odległości będą poprawne po h turach, gdzie h to średnica sieci.
- Informacja o dodaniu routera lub łącza propaguje się z prędkością jednej krawędzi na turę.
- A informacja o awarii?

Awaria łącza

- Jeśli niedostępna staje się sieć S podłączona bezpośrednio:
  - $d(X, S) \leftarrow \infty$
- Jeśli dowiadujesz się o (dowolnej) odległości od routera A, który jest pierwszym routerem na trasie do sieci S:
  - Aktualizuj, nawet jeśli nowa trasa jest **gorsza niż obecna**.
  - $d(X, S) \leftarrow d(X, S_A) + d(A, S)$

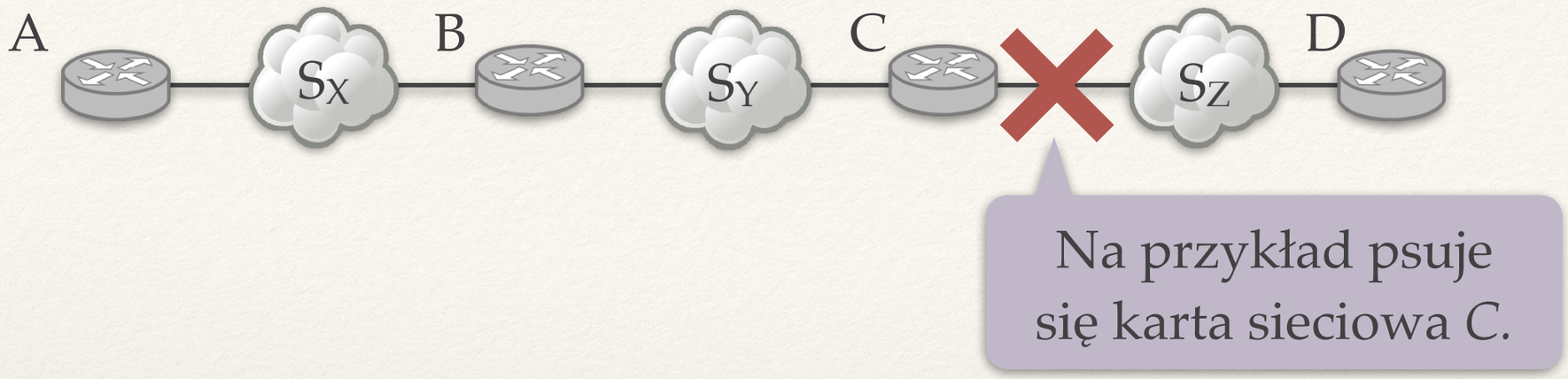
A jest sąsiadem X leżącym w sieci S_A odległej o $d(X, S_A)$.

A mówi: „mam do *S* odległość $d(A,S)$“.

Przykład awarii: psuje się połączenie C-D (1)

trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1

Przykład awarii: psuje się połączenie C-D (1)



trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞

Przykład awarii: psuje się połączenie C-D (1)

Dobry przypadek:

- C przekazuje swoją tablicę do B (wcześniej niż B do C).

trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞
czas 2	3 (via B)	∞	∞

Przykład awarii: psuje się połączenie C-D (1)

Dobry przypadek:

- C przekazuje swoją tablicę do B (wcześniej niż B do C).
- B przekazuje swoją tablicę do A (wcześniej niż A do B).

trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞
czas 2	3 (via B)	∞	∞
czas 3	∞	∞	∞

Przykład awarii: psuje się połączenie C-D (2)

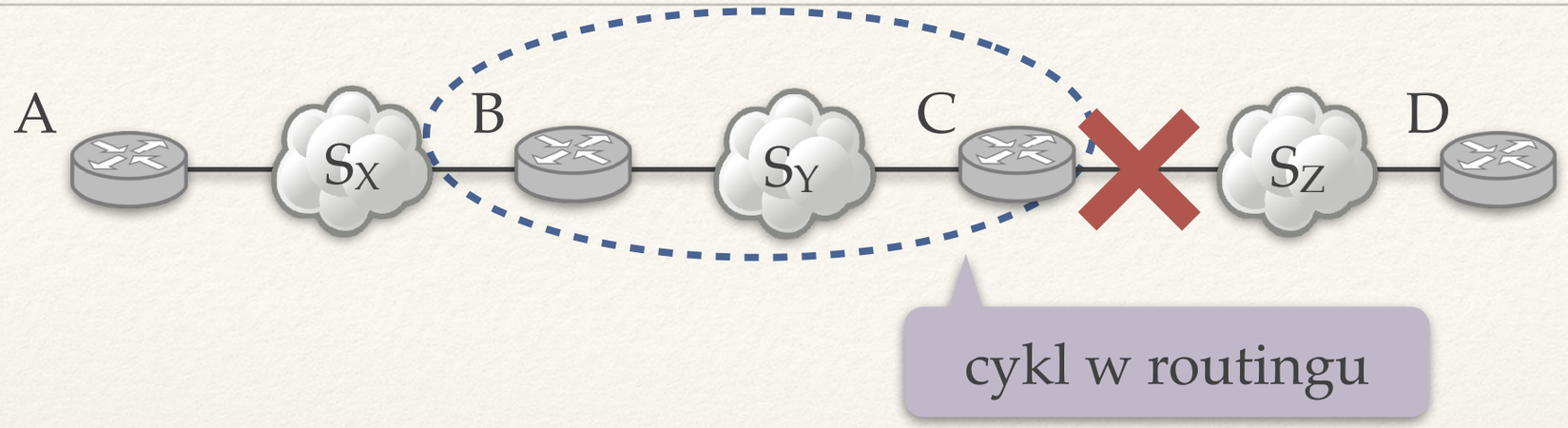
trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞

Przykład awarii: psuje się połączenie C-D (2)

Zły przypadek: B przekazuje swoją tablicę do C (wcześniej niż C do B).

trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞
czas 2	3 (via B)	2 (via C)	**3 (via B)**

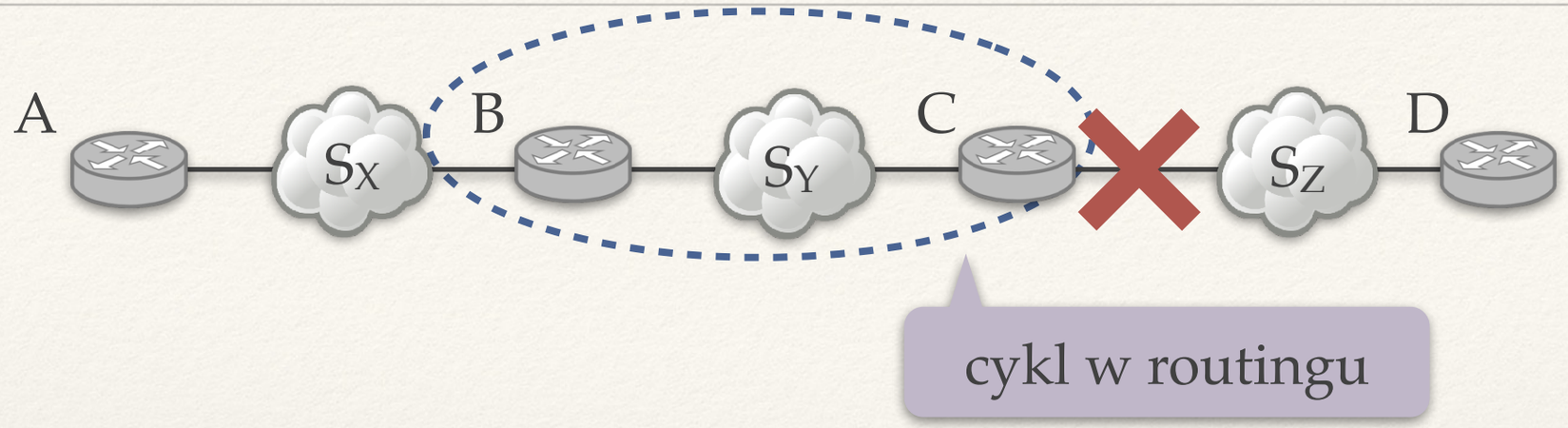
Przykład awarii: psuje się połączenie C-D (2)



Zły przypadek: B przekazuje swoją tablicę do C (wcześniej niż C do B).

trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞
czas 2	3 (via B)	2 (via C)	**3 (via B)**

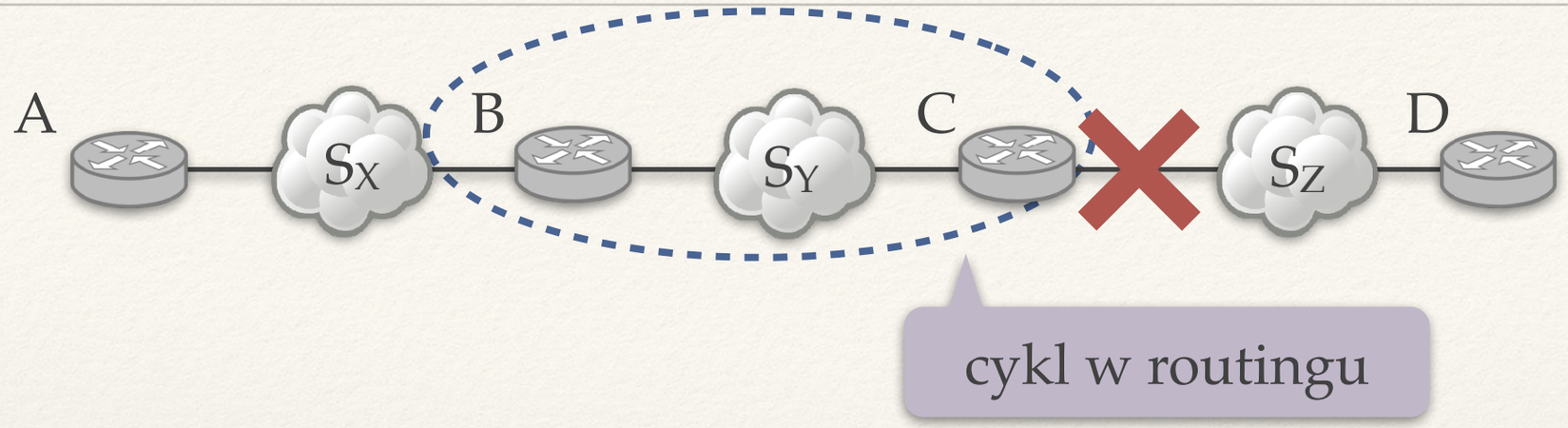
Przykład awarii: psuje się połączenie C-D (2)



Zły przypadek: B przekazuje swoją tablicę do C (wcześniej niż C do B).

trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞
czas 2	3 (via B)	2 (via C)	**3 (via B)**
czas 3	3 (via B)	4 (via C)	3 (via B)

Przykład awarii: psuje się połączenie C-D (2)



Zły przypadek: B przekazuje swoją tablicę do C (wcześniej niż C do B).

trasa do S_Z	A	B	C
czas 0	3 (via B)	2 (via C)	1
czas 1	3 (via B)	2 (via C)	∞
czas 2	3 (via B)	2 (via C)	**3 (via B)**
czas 3	3 (via B)	4 (via C)	3 (via B)
czas 4	5 (via B)	4 (via C)	5 (via B)
czas 5	5 (via B)	6 (via C)	5 (via B)
…	…	…	…

Zliczanie do nieskończoności (1)

- Routery zwiększają znaną odległość do S_Z średnio o 1 na turę.
- *B* wysłał do *C* informację o odległości do S_Z, ale *C* jest na tej trasie!
- Zatruwanie ścieżki zwrotnej (*poison reverse*):
  - Jeśli router *X* jest wpisany jako następny router na ścieżce do *S*, to wysyłamy do *X* informację „mam do *S* ścieżkę nieskończoną“.
  - Może nie pomóc w większych sieciach (ćwiczenie).

Zliczanie do nieskończoności (2)

Dodatkowe heurystyki:

- Wysyłanie również pierwszego routera na trasie (nie pomaga w większych sieciach).
- Szybsza aktualizacja w momencie wykrycia awarii.
- Jeśli wszystko inne zawiedzie: ustalić wartość graniczną odległości, powyżej której router jest uważany za nieosiągalny.

Protokół RIP (Routing Information Protocol)

- Implementacja algorytmu wektora odległości.
- Wysyłanie wektora odległości co 30 sek + w momencie zmiany.
- Zatruwanie ścieżki zwrotnej.
- $\infty = 16$ (w RIPv1), $\infty = 256$ (w RIPv2).
- Nieefektywny dla większych sieci.

Porównanie algorytmów

	stan łączy	wektory odległości
pamięć	O(|V|+|E|)	O(|V|)
implementacja	trudniejsza (zalewanie)	łatwiejsza (kontakt z sąsiadami)
szybkość zbieżności (w praktyce)	szybsza	wolniejsza
zapotrzebowanie na moc obliczeniową	większe (algorytm Dijkstry)	mniejsze (aktualizacja odległości)

Routing w Internecie

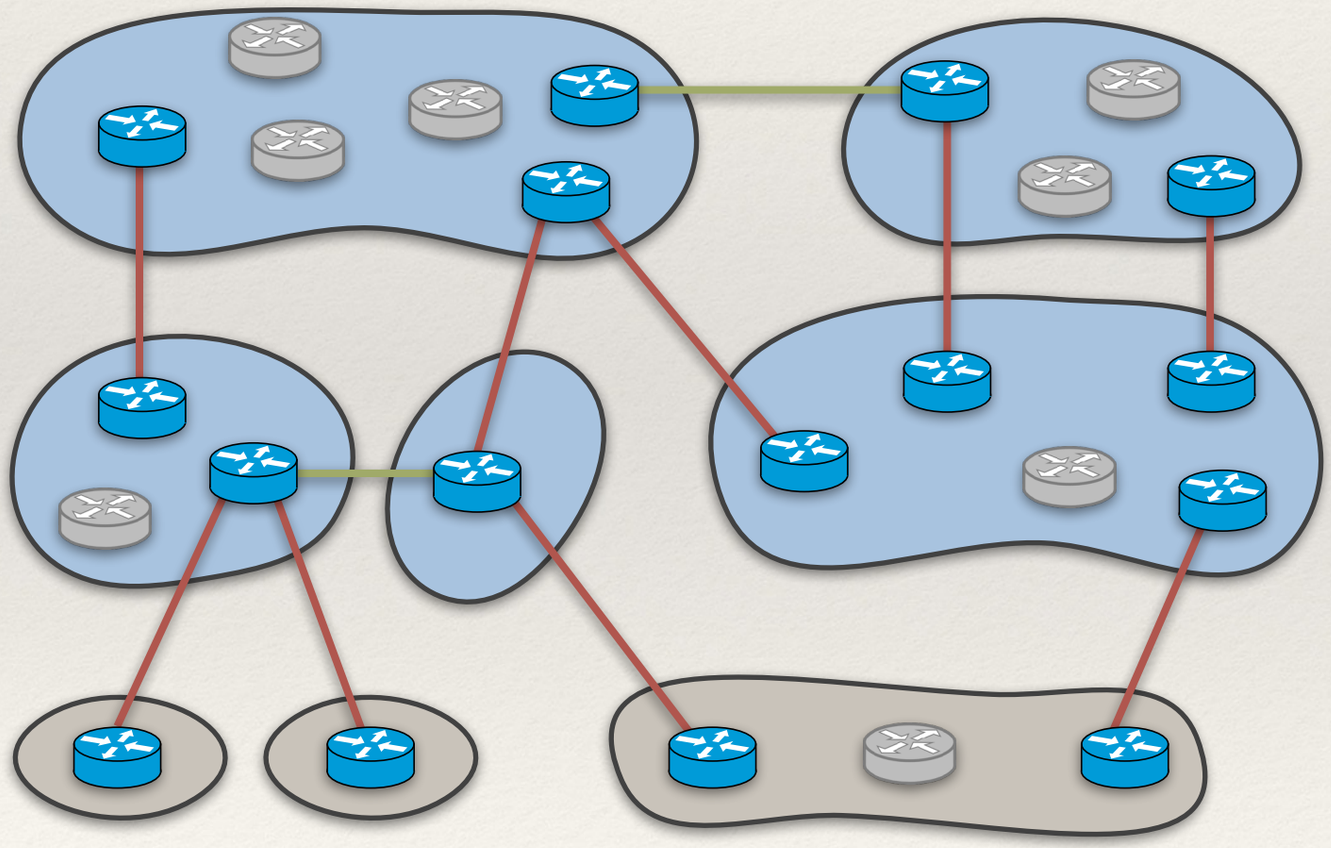
Routing w Internecie

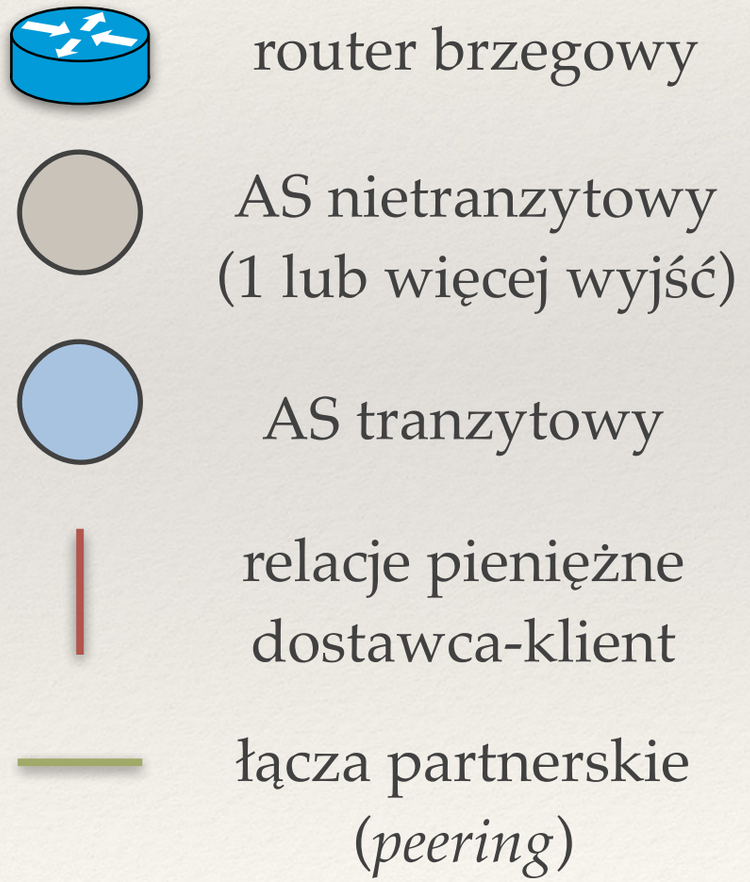
- Omówiliśmy dwa podejścia do routingu.
- Te podejścia nie skalują się do całego Internetu.
- Wykorzystywane są **wewnątrz** systemów autonomicznych.

Systemy autonomiczne (AS)

Każdy ISP posiada jeden lub więcej AS.

- ~20 tys. ISP, ~100 tys. AS.
- Spójna polityka wewnętrznego routingu (często OSPF, rzadziej RIP).



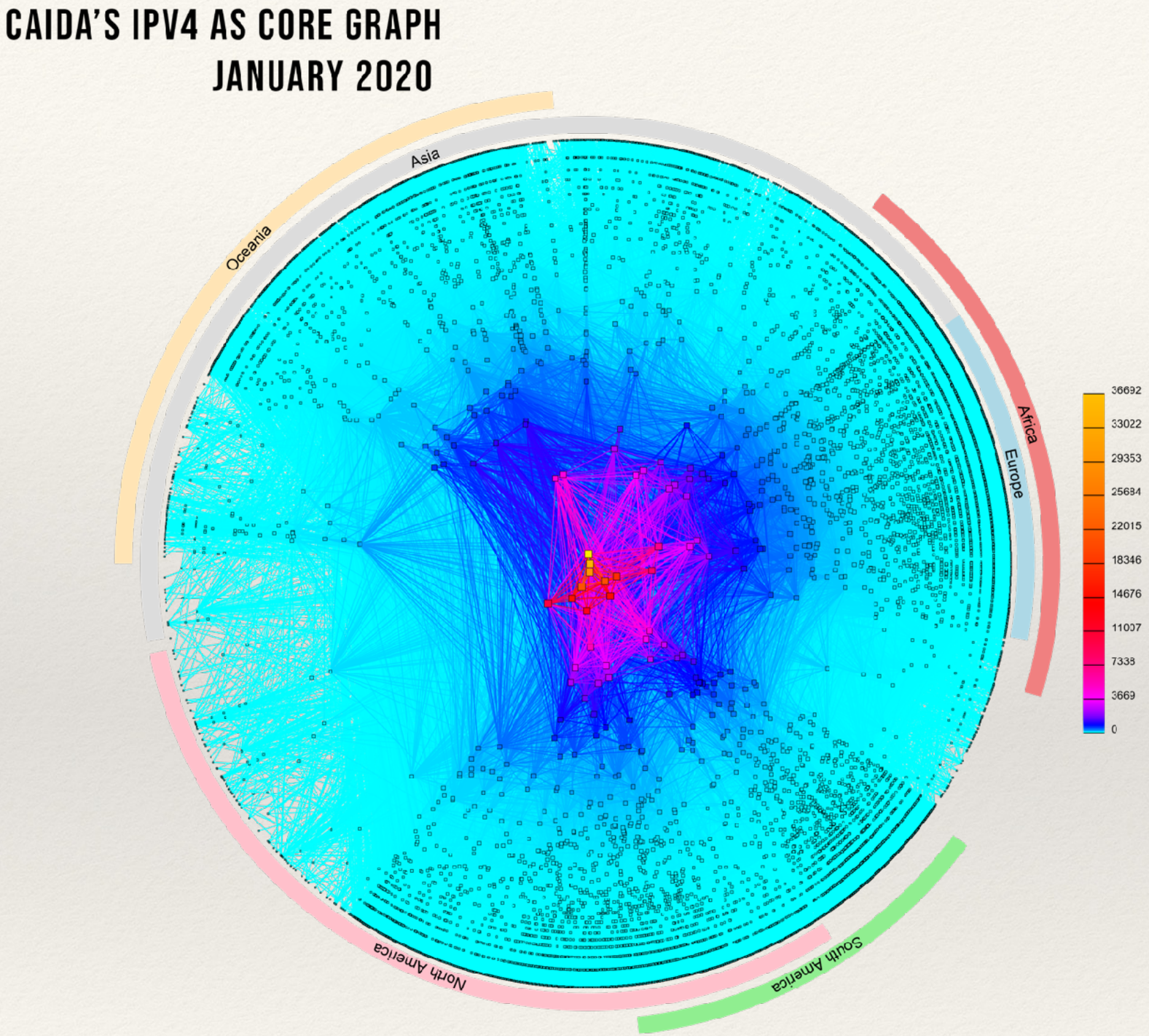


Przykładowa trasa wraz z AS

```
$> traceroute -A google.com

traceroute to google.com (172.217.20.206), 30 hops max
 1 172.16.16.254 (172.16.16.254) [*]
 2 info.wask.wroc.pl (156.17.4.254) [AS8970]
 3 matchem-vprn509-curie-uni.wask.wroc.pl (156.17.252.26) [AS8970]
 4 uwrvprn509-unir2.wask.wroc.pl (156.17.252.37) [AS8970]
 5 unir2-uwrvprn509.wask.wroc.pl (156.17.252.36) [AS8970]
 6 z-Wroclaw.lodz-gw2.10Gb.rtr.pionier.gov.pl (212.191.240.121) [AS8501]
 7 poznan-gw3.z-lodz-gw2.rtr.pionier.gov.pl (212.191.126.70) [AS8501]
 8 72.14.203.178 (72.14.203.178) [AS15169]
 9 108.170.250.209 (108.170.250.209) [AS15169]
10 216.239.41.171 (216.239.41.171) [AS15169]
   216.239.41.169 (216.239.41.169) [AS15169]
11 waw02s08-in-f14.1e100.net (172.217.20.206) [AS15169]
```

Mapa ISP z 2020 roku



Obrazek ze strony
https://www.caida.org/projects/cartography/as-core/2020/

Czego chcą ISP?

- Wybór tras routingu na podstawie **polityki ISP**, np.:
  - „Chcę płacić jak najmniej“.
  - „Nie chcę udostępniać wewnętrznych szczegółów na temat AS“.
  - „Nie chcę żeby ktoś przesyłał dane przez mój AS, jeśli nie mam z tego zysku“.
- Względy ekonomiczne, prywatności, autonomii.
- Polityki nie są realizowane przez najkrótsze ścieżki.

BGP (Border Gateway Protocol)

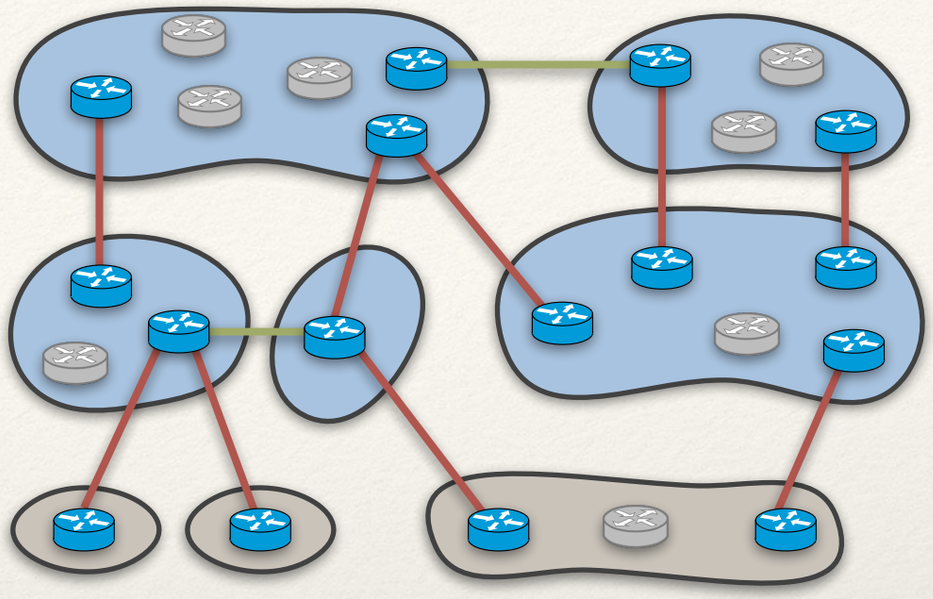
Algorytm routingu pomiędzy AS.

- Bazuje na algorytmach wektora odległości.
  - Algorytmy stanu łączy nie gwarantują prywatności.
- Rozgłaszane są całe poznane trasy *„Sieć 123.123.0.0/16 jest osiągalna przez trasę {AS3, AS21, AS13}, pierwszy router to 34.34.34.34“.*
  → Łatwe unikanie cykli.
- ISP sam decyduje:
  - czy i komu rozgłosić poznaną trasę;
  - które trasy wykorzystać do tworzenia tablic przekazywania.

Które trasy rozgłaszać?

- **Zawartość naszego AS (prefiksy CIDR):**
  - Żeby Internet mógł do nas trafić.

- **Trasy do naszych klientów:**
  - Żeby Internet mógł do nich trafić.
  - Tak, bo płacą nam za przesyłane dane.



- **Trasy do naszych dostawców:**
  - Naszym klientom tak.
  - Poza tym nie: nie chcemy, żeby inni przesyłali przez nasz AS dane do naszego dostawcy (my płacimy, ale nam nie płacą).

- **Trasy do naszych partnerów:**
  - Naszym klientom tak.

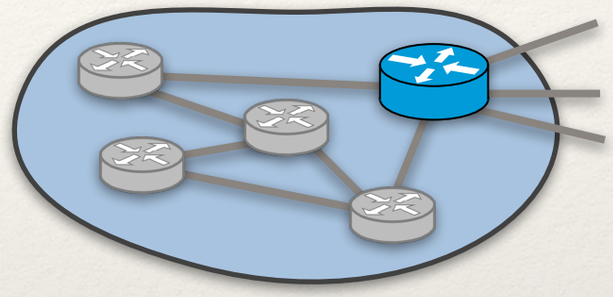
Wybór tras

Wiele możliwości dotarcia do jakiejś sieci (prefiksu CIDR).

- Zazwyczaj wybór najkrótszej trasy (najmniejsza liczba AS).
- Ale często modyfikowany przez politykę:
  - wybierz najpierw trasę przez swojego klienta,
  - … potem przez partnera,
  - … a na końcu trasę przez dostawcę.

Współpraca BGP z routingiem wewnątrz AS (1)

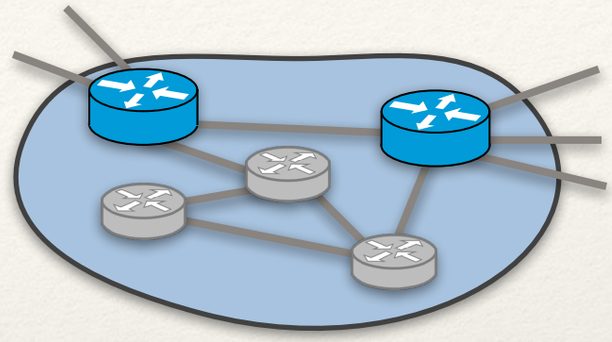
- **Routery brzegowe X_i danego AS (via BGP):**
  - rozgłaszają prefiksy CIDR tego AS;
  - dowiadują się o trasach do innych AS.



- **AS z jednym wyjściem *X*.**
  - Oblicz najkrótsze ścieżki wewnątrz AS.
  - Dodaj *X* na wszystkich routerach wewnątrz AS jako bramę domyślną.

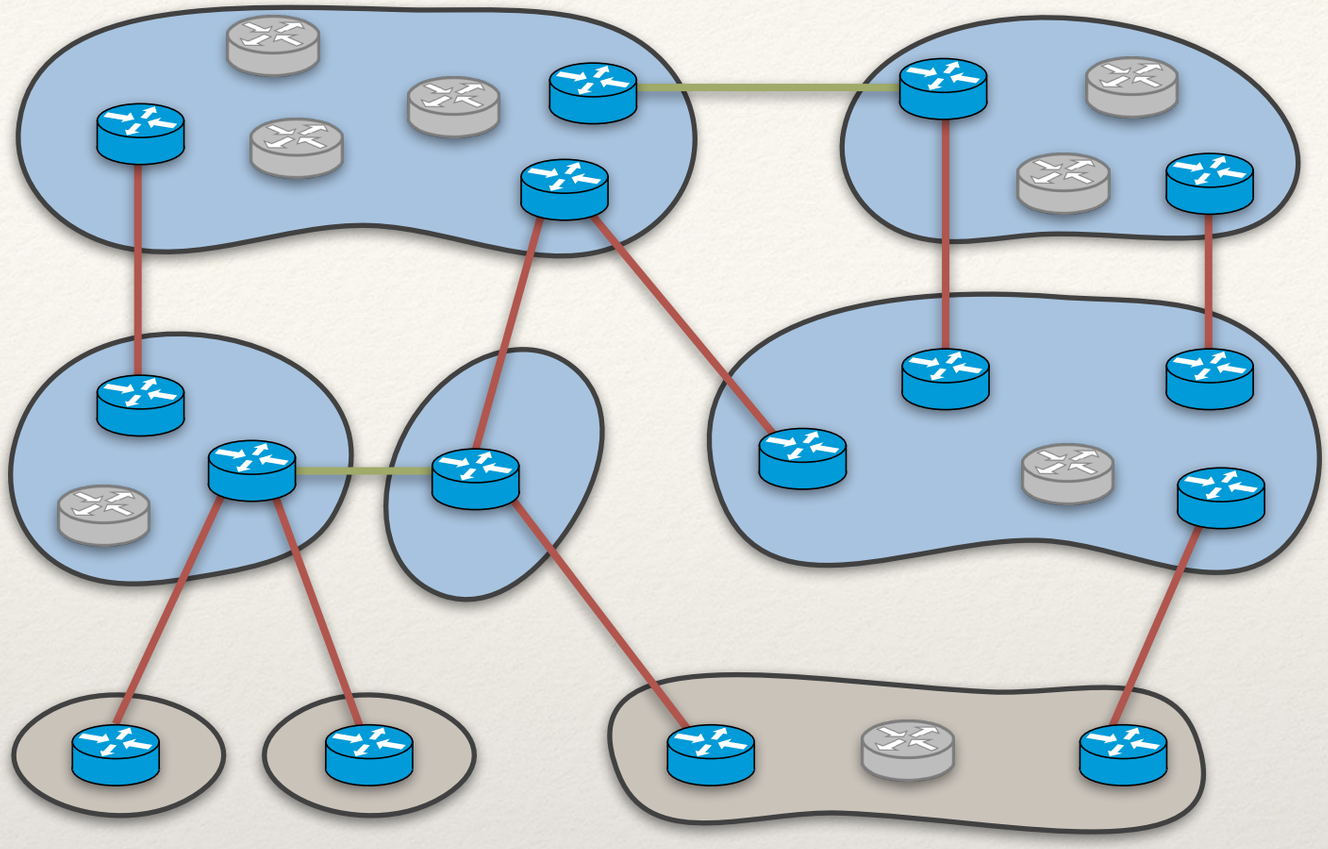
Współpraca BGP z routingiem wewnątrz AS (2)

- **Routery brzegowe X_i danego AS (via BGP):**
  - rozgłaszają prefiksy CIDR tego AS;
  - dowiadują się o trasach do innych AS.



- **AS z wieloma wyjściami X_1, X_2, X_3,….**
  - Protokół routingu wewnątrz AS: X_i udostępniają wiedzę z BGP jako swoje „sąsiedztwo“ (z odpowiednimi odległościami).
    - Wada: każdy router wewnątrz AS przechowuje informacje o sieciach z całego Internetu
    - Są lepsze rozwiązania (iBGP).

Czego brakuje na obrazku



- **IXP** (*Internet Exchange Point*): punkt wymiany ruchu, łączy ze sobą wiele routerów brzegowych, często w relacji peering.
- **CDN** (*Content Delivery Networks*): jak AS, ale celem jest dostarczanie treści jak najbliżej użytkowników końcowych (Akamai, Cloudflare, Google, Netflix, …).

Lektura dodatkowa

- Kurose & Ross: rozdział 5.
- Tanenbaum: rozdział 5.

- Dokumentacja RIP i OSPF:
  - https://web.archive.org/web/20211023182600/http://www.networksorcery.com/enp/protocol/rip.htm
  - https://web.archive.org/web/20211025013604/http://networksorcery.com/enp/protocol/ospf.htm

- Różne ciekawostki:
  - Jak ekonomia ukształtowała BGP: http://web.mit.edu/6.829/www/currentsemester/papers/AS-bgp-notes.pdf
  - Jak Pakistan przejął YouTube: https://www.youtube.com/watch?v=IzLPKuAOe50

Zagadnienia

- Co to jest cykl w routingu? Co go powoduje?
- Czym różni się tablica routingu od tablicy przekazywania?
- Dlaczego w algorytmach routingu dynamicznego obliczamy najkrótsze ścieżki?
- Co to jest metryka? Jakie metryki mają sens?
- Czym różnią się algorytmy wektora odległości od algorytmów stanów łączy?
- Jak router może stwierdzić, że bezpośrednio podłączona sieć jest nieosiągalna?
- Co to znaczy, że stan tablic routingu jest stabilny?
- Jak zalewać sieć informacją? Co to są komunikaty LSA?
- Co wchodzi w skład wektora odległości?
- W jaki sposób podczas działania algorytmu routingu dynamicznego może powstać cykl w routingu?
- Co to jest problem zliczania do nieskończoności? Kiedy występuje?
- Na czym polega technika zatruwania ścieżki zwrotnej (*poison reverse*)?
- Po co w algorytmach wektora odległości definiuje się największą odległość w sieci (16 w protokole RIPv1)?
- Co to jest system autonomiczny (AS)? Jakie znasz typy AS?
- Czym różnią się połączenia dostawca-klient pomiędzy AS od łącz partnerskich (*peering*)?
- Dlaczego w routingu pomiędzy systemami autonomicznymi nie stosuje się najkrótszych ścieżek?
- Które trasy w BGP warto rozgłaszać i komu? A które wybierać?
- Jak BGP może współpracować z algorytmami routingu wewnątrz AS?