Ćwiczenia 1

Sieci Komputerowe

Wojciech Adamiec

23 marca 2021

Zadanie	1	2	3	4	5	6	7	8	9	10
Rozwiązanie	X	X	X	X	X	X	X	X	X	X

Spis treści

1. Zadanie

1. Dla każdego z podanych poniżej adresów IP w notacji CIDR określ, czy jest to adres sieci, adres rozgłoszeniowy czy też adres komputera. W każdym przypadku wyznacz odpowiadający mu adres sieci, rozgłoszeniowy i jakiś adres IP innego komputera w tej samej sieci.

   - `10.1.2.3/8`
   - `156.17.0.0/16`
   - `99.99.99.99/27`
   - `156.17.64.4/30`
   - `123.123.123.123/32`

1.1. 10.1.2.3/8

1. Adres sieci: 10.0.0.0/8
2. Adres rozgłoszeniowy: 10.255.255.255
3. Dowolny adres komputera: 10.0.0.1

1.2. 156.17.0.0/16

1. Adres sieci: 156.17.0.0/16
2. Adres rozgłoszeniowy: 156.17.255.255
3. Dowolny adres komputera: 156.17.0.1

1.3. 99.99.99.99/27

1. Adres sieci: 99.99.99.96/27
2. Adres rozgłoszeniowy: 99.99.99.127
3. Dowolny adres komputera: 99.99.99.97

1.4. 156.17.64.4/30

1. Adres sieci: 156.17.64.4/30
2. Adres rozgłoszeniowy: 156.17.64.7
3. Dowolny adres komputera: 156.17.64.5

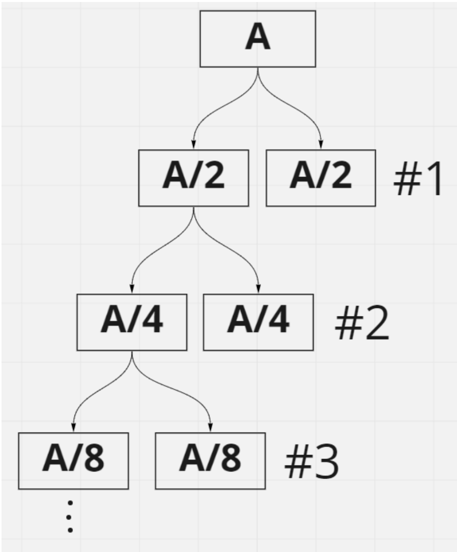
1.5. 123.123.123.123/32

1. Adres sieci: 123.123.123.123/32
2. Adres rozgłoszeniowy: 123.123.123.123
3. Jedyny adres komputera: 123.123.123.123

2. Zadanie

2. Podziel sieć `10.10.0.0/16` na 5 rozłącznych podsieci, tak aby każdy z adresów IP z sieci `10.10.0.0/16` był w jednej z tych 5 podsieci. Jak zmieniła się liczba adresów IP możliwych do użycia przy adresowaniu komputerów? Jaki jest minimalny rozmiar podsieci, który możesz uzyskać w ten sposób?

Pomysł na podział jest następujący: Dzielimy naszą sieć na pół (ustalamy dla każdej połówki inny stan pierwszego bitu z lewej), następnie jedną z podsieci ponownie dzielimy na pół i tak dalej.



W ten sposób jesteśmy w stanie podzielić naszą sieć na 5 rozłącznych podsieci, tak, że nie tracimy żadnych adresów. Precyzując:

1. X.X.00000000.00000000 - ustalone 0 z przodu; mamy 15 wolnych bitów
   10.10.0.0/17
2. X.X.10000000.00000000 - ustalone 10 z przodu; mamy 14 wolnych bitów
   10.10.128.0/18
3. X.X.11000000.00000000 - ustalone 110 z przodu; mamy 13 wolnych bitów
   10.10.192.0/19
4. X.X.11100000.00000000 - ustalone 1110 z przodu; mamy 12 wolnych bitów
   10.10.224.0/20
5. X.X.11110000.00000000 - ustalone 1111 z przodu; mamy 12 wolnych bitów
   10.10.240.0/20

Jak zmieniła się liczba adresów? Otóż zamiast jednego adresu rozgłoszeniowego i jednego adresu sieci mamy ich teraz po 5. Zatem sumarycznie straciliśmy 8 adresów.

Jaki jest minimalny rozmiar podsieci jaki możemy w ten sposób osiągnąć? Otóż dla warunków z naszego zadania najmniejsza podsieć będzie mieć rozmiar 2^{12}.

3. Zadanie

3. Tablica routingu zawiera następujące wpisy (podsieć $\to$ dokąd wysłać):

- `0.0.0.0/0` $\to$ do routera A
- `10.0.0.0/23` $\to$ do routera B
- `10.0.2.0/24` $\to$ do routera B
- `10.0.3.0/24` $\to$ do routera B
- `10.0.1.0/24` $\to$ do routera C
- `10.0.0.128/25` $\to$ do routera B
- `10.0.1.8/29` $\to$ do routera B
- `10.0.1.16/29` $\to$ do routera B
- `10.0.1.24/29` $\to$ do routera B

Napisz równoważną tablicę routingu zawierającą jak najmniej wpisów.

Nr	Adresy	Kierunek	Zakres
1	0.0.0.0/0	A	0.0.0.0 - 255.255.255.255
2	10.0.0.0/23	B	10.0.0.0 - 10.0.1.255
3	10.0.2.0/24	B	10.0.2.0 - 10.0.2.255
4	10.0.3.0/24	B	10.0.3.0 - 10.0.3.255
5	10.0.1.0/24	C	10.0.1.0 - 10.0.1.255
6	10.0.0.128/25	B	10.0.0.128 - 10.0.0.255
7	10.0.1.8/29	B	10.0.1.8 - 10.0.1.15
8	10.0.1.16/29	B	10.0.1.16 - 10.0.1.23
9	10.0.1.24/29	B	10.0.1.24 - 10.0.1.31

Zacznijmy od spostrzeżenia, że możemy połączyć ze sobą zakresy 2, 3 i 4. Powstanie wtedy jeden zakres I. Nie trudno zauważyć, że teraz zakres 6 jest zaledwie podzakresem naszego nowo utworzonego zakresu I (oraz jest rozłączny z zakresem prowadzącym do C) - można go usunąć.

Podobnie możemy połączyć ze sobą zakresy 8 i 9. Powstanie na ich miejsce zakres II.

W sumie dzięki tym uproszczeniom dostajemy taką tablicę:

Nr	Adresy	Kierunek	Zakres
1	0.0.0.0/0	A	0.0.0.0 - 255.255.255.255
I	10.0.0.0/22	B	10.0.0.0 - 10.0.3.255
5	10.0.1.0/24	C	10.0.1.0 - 10.0.1.255
7	10.0.1.8/29	B	10.0.1.8 - 10.0.1.15
II	10.0.1.16/28	B	10.0.1.16 - 10.0.1.31

4. Zadanie

4. Wykonaj powyższe zadanie dla tablicy

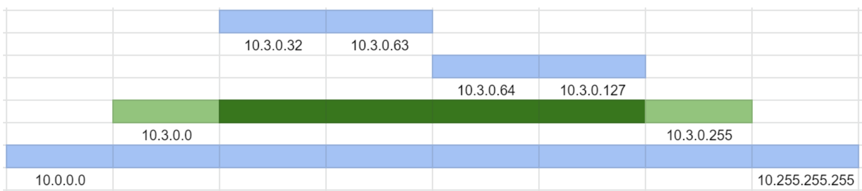
- `0.0.0.0/0` → do routera *A*
- `10.0.0.0/8` → do routera *B*
- `10.3.0.0/24` → do routera *C*
- `10.3.0.32/27` → do routera *B*
- `10.3.0.64/27` → do routera *B*
- `10.3.0.96/27` → do routera *B*

Nr	Adresy	Kierunek	Zakres
1	0.0.0.0/0	A	0.0.0.0 - 255.255.255.255
2	10.0.0.0/8	B	10.0.0.0 - 10.255.255.255
3	10.3.0.0/24	C	10.3.0.0 - 10.3.0.255
4	10.3.0.32/27	B	10.3.0.32 - 10.3.0.63
5	10.3.0.64/27	B	10.3.0.64 - 10.3.0.95
6	10.3.0.96/27	B	10.3.0.96 - 10.3.0.127

Zakresy 5 i 6 możemy ze sobą połączyć w jeden zakres *I*. Dzięki temu dostajemy taką tablicę:

Nr	Adresy	Kierunek	Zakres
1	0.0.0.0/0	A	0.0.0.0 - 255.255.255.255
2	10.0.0.0/8	B	10.0.0.0 - 10.255.255.255
3	10.3.0.0/24	C	10.3.0.0 - 10.3.0.255
4	10.3.0.32/27	B	10.3.0.32 - 10.3.0.63
I	10.3.0.64/26	B	10.3.0.64 - 10.3.0.127

Graficznie w tym momencie mamy taką sytuację:



Chcemy spróbować redukcji liczby zakresów. W tym celu musimy sprawdzić czy da się jakoś odseparować od siebie *nieprzykryte* części zakresu 3. Okazuje się, że można to zrobić.

Nr	Adresy	Kierunek	Zakres
1	0.0.0.0/0	A	0.0.0.0 - 255.255.255.255
2	10.0.0.0/8	B	10.0.0.0 - 10.255.255.255
II	10.3.0.0/27	C	10.3.0.0 - 10.3.0.31
III	10.3.0.128/25	C	10.3.0.128 - 10.3.0.255

5. Zadanie

5. Jak uporządkować wpisy w tablicy routingu, żeby zasada najlepszego dopasowania odpowiadała wyborowi „pierwszy pasujący" (tj. przeglądaniu tablicy od początku do końca aż do momentu napotkania dowolnej pasującej reguły)? Odpowiedź uzasadnij formalnie.

Twierdzenie:
Wybór *pierwszy pasujący* spełnia zasadę najlepszego dopasowania, jeśli wpisy w tablicy routingu są posortowane malejąco względem długości prefiksu.

Dowód:
Weźmy dowolną tablicę routingu, załóżmy, że wpisy w niej są posortowane malejąco względem długości prefiksu.

Pokażemy, że dla dowolnego adresu IP pierwszy pasujący wpis spełnia zasadę najlepszego dopasowania.

Weźmy dowolny adres IP, nazwijmy go ip. Załóżmy, że pierwszy napotkany w naszej tablicy wpis, do którego można dopasować ip nazywa się k.

Niech n będzie długością prefiksu k. Wiemy, że pierwsze n bitów adresu ip jest identyczne jak we wpisie k.

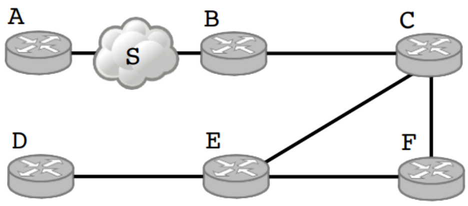
Rozpatrzmy teraz dowolny wpis s zapisany w naszej tablicy po wpisie k. Niech długość jego prefiksu będzie równa m. Dzięki temu, że nasza tablica jest posortowana wiemy, że $m \leq n$.

Oznacza to, że dopasowane zostanie nie więcej bitów adresu ip niż n. To z kolei oznacza, że dopasowanie ip z s jest nie lepsze od dopasowania ip z k.

Wnioskujemy z tego, że dopasowanie ip z k jest najlepszym dopasowaniem. Oznacza to, że dla posortowanej malejąco względem długości prefiksu tablicy routingu pierwszy pasujący wpis jest jednocześnie tym najlepiej dopasowanym - co należało dowieść.

6. Zadanie

6. W podanej niżej sieci tablice routingu budowane są za pomocą algorytmu wektora odległości. Pokaż (krok po kroku), jak będzie się to odbywać. W ilu krokach zostanie osiągnięty stan stabilny?



Zaczynamy od bezpośrednich połączeń między routerami.

	A	B	C	D	E	F
Trasa do A	---	1				
Trasa do B	1	---	1			
Trasa do C		1	---		1	1
Trasa do D				---	1	
Trasa do E			1	1	---	1
Trasa do F			1		1	---
Trasa do S	1	1				

W każdym kroku będziemy przesyłać informacje o ścieżkach jeden skok dalej. Jeśli nowa ścieżka jest krótsza od aktualnej to odpowiednie pole w tablicy zostanie zaktualizowane.

	A	B	C	D	E	F
Trasa do A	---	1	2 (B)			
Trasa do B	1	---	1		2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D			2 (E)	---	1	2 (E)
Trasa do E		2 (C)	1	1	---	1
Trasa do F		2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)			

	A	B	C	D	E	F
Trasa do A	---	1	2 (B)		3 (C)	3 (C)
Trasa do B	1	---	1	3 (E)	2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D		3 (C)	2 (E)	---	1	2 (E)
Trasa do E	3 (B)	2 (C)	1	1	---	1
Trasa do F	3 (B)	2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)		3 (C)	3 (C)

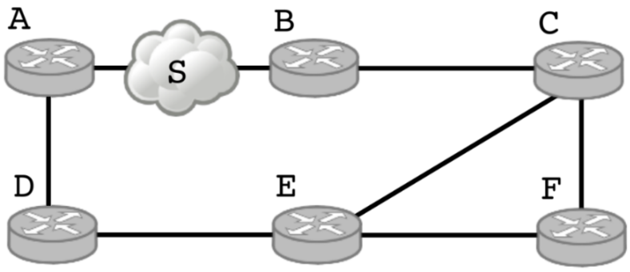
	A	B	C	D	E	F
Trasa do A	---	1	2 (B)	4 (E)	3 (C)	3 (C)
Trasa do B	1	---	1	3 (E)	2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D	4 (B)	3 (C)	2 (E)	---	1	2 (E)
Trasa do E	3 (B)	2 (C)	1	1	---	1
Trasa do F	3 (B)	2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)	4 (E)	3 (C)	3 (C)

	A	B	C	D	E	F
Trasa do A	---	1	2 (B)	4 (E)	3 (C)	3 (C)
Trasa do B	1	---	1	3 (E)	2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D	4 (B)	3 (C)	2 (E)	---	1	2 (E)
Trasa do E	3 (B)	2 (C)	1	1	---	1
Trasa do F	3 (B)	2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)	4 (E)	3 (C)	3 (C)

Po 3 krokach dostajemy już stan stabilny.

7. Zadanie

7. Załóżmy, że w powyższej sieci tablice routingu zostały już zbudowane. Co będzie się działo, jeśli zostanie dodane połączenie między routerami *A* i *D*?



Zaczynamy od stanu stabilnego z końca poprzedniego zadania.

	A	B	C	D	E	F
Trasa do A	---	1	2 (B)	4 (E)	3 (C)	3 (C)
Trasa do B	1	---	1	3 (E)	2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D	4 (B)	3 (C)	2 (E)	---	1	2 (E)
Trasa do E	3 (B)	2 (C)	1	1	---	1
Trasa do F	3 (B)	2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)	4 (E)	3 (C)	3 (C)

Następnie dodajemy nowe połączenie i rozpoczynamy na nowo działanie naszego algorytmu.

	A	B	C	D	E	F
Trasa do A	---	1	2 (B)	1	3 (C)	3 (C)
Trasa do B	1	---	1	3 (E)	2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D	1	3 (C)	2 (E)	---	1	2 (E)
Trasa do E	3 (B)	2 (C)	1	1	---	1
Trasa do F	3 (B)	2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)	4 (E)	3 (C)	3 (C)

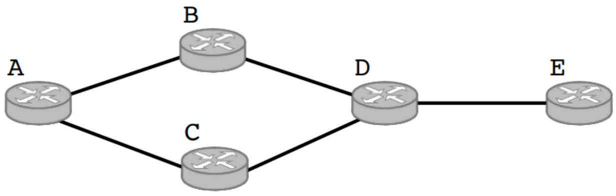
	A	B	C	D	E	F
Trasa do A	---	1	2 (B)	1	2 (D)	3 (C)
Trasa do B	1	---	1	2 (A)	2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D	1	2 (A)	2 (E)	---	1	2 (E)
Trasa do E	2 (D)	2 (C)	1	1	---	1
Trasa do F	3 (B)	2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)	2 (A)	3 (C)	3 (C)

Ponownie dostajemy stan stabilny.

	A	B	C	D	E	F
Trasa do A	---	1	2 (B)	1	2 (D)	3 (C)
Trasa do B	1	---	1	2 (A)	2 (C)	2 (C)
Trasa do C	2 (B)	1	---	2 (E)	1	1
Trasa do D	1	2 (A)	2 (E)	---	1	2 (E)
Trasa do E	2 (D)	2 (C)	1	1	---	1
Trasa do F	3 (B)	2 (C)	1	2 (E)	1	---
Trasa do S	1	1	2 (B)	2 (A)	3 (C)	3 (C)

8. Zadanie

8. W przedstawionej poniżej sieci uszkodzeniu ulega połączenie między routerami D i E. Załóżmy, że w sieci działa algorytm wektora odległości wykorzystujący technikę zatruwania ścieżki zwrotnej (*poison reverse*). Pokaż — opisując krok po kroku jakie komunikaty są przesyłane między routerami — że może powstać cykl w routingu.



Zaczynamy od stanu stabilnego. Dochodzi do przerwania połączenia między D i E. Informacja o tej awarii zaczyna rozchodzić się po sieci.

	A	B	C	D	E
Trasa do A	---	1	1	2(B)	3(D)
Trasa do B	1	---	2(A)	1	2(D)
Trasa do C	1	2(A)	---	1	2(D)
Trasa do D	2(C)	1	1	---	1
Trasa do E	3(C)	2(D)	2(D)	1	---

	A	B	C	D	E
Trasa do A	---	1	1	2(B)	inf
Trasa do B	1	---	2(A)	1	inf
Trasa do C	1	2(A)	---	1	inf
Trasa do D	2(C)	1	1	---	inf
Trasa do E	3(C)	2(D)	2(D)	inf	---

	A	B	C	D	E
Trasa do A	---	1	1	2(B)	inf
Trasa do B	1	---	2(A)	1	inf
Trasa do C	1	2(A)	---	1	inf
Trasa do D	2(C)	1	1	---	inf
Trasa do E	3(C)	inf	inf	inf	---

Następnie może dojść do poniższej patologii, której zabezpieczenie w postaci *poison reverse* nie zapobiegnie.

	A	B	C	D	E
Trasa do A	---	1	1	2(B)	inf
Trasa do B	1	---	2(A)	1	inf
Trasa do C	1	2(A)	---	1	inf
Trasa do D	2(C)	1	1	---	inf
Trasa do E	3(C)	4(A)	inf	inf	---

	A	B	C	D	E
Trasa do A	---	1	1	2(B)	inf
Trasa do B	1	---	2(A)	1	inf
Trasa do C	1	2(A)	---	1	inf
Trasa do D	2(C)	1	1	---	inf
Trasa do E	3(C)	4(A)	inf	5(B)	---

	A	B	C	D	E
Trasa do A	---	1	1	2(B)	inf
Trasa do B	1	---	2(A)	1	inf
Trasa do C	1	2(A)	---	1	inf
Trasa do D	2(C)	1	1	---	inf
Trasa do E	3(C)	4(A)	6(D)	5(B)	---

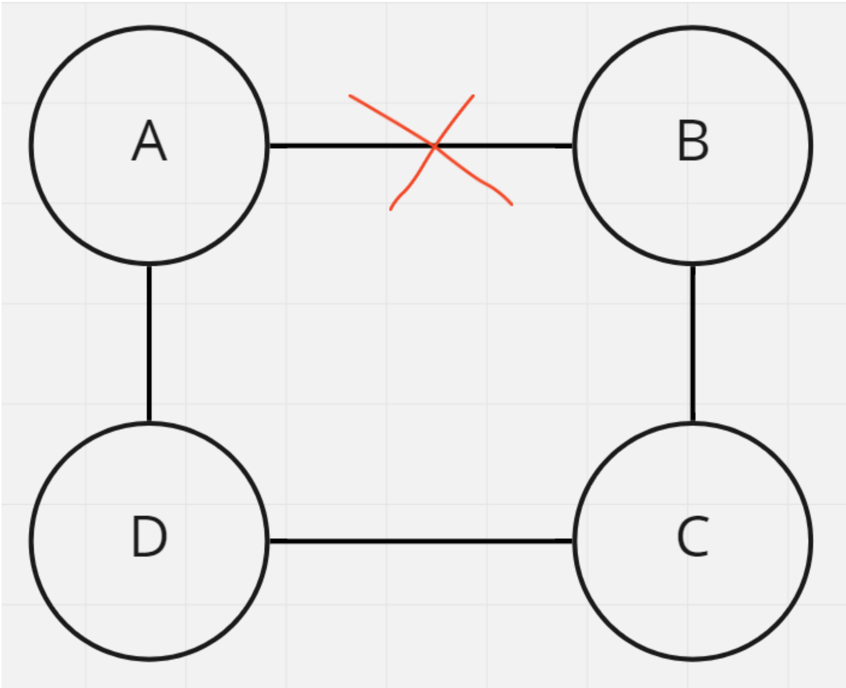
	A	B	C	D	E
Trasa do A	---	1	1	2(B)	inf
Trasa do B	1	---	2(A)	1	inf
Trasa do C	1	2(A)	---	1	inf
Trasa do D	2(C)	1	1	---	inf
Trasa do E	3(C)	4(A)	6(D)	5(B)	---

Co zakończy się powstaniem cyklu.

9. Zadanie

9. Pokaż, że przy wykorzystaniu algorytmu stanu łączy też może powstać cykl w routingu. W tym celu skonstruuj topologię sieci z dwoma wyróżnionymi, bezpośrednio połączonymi routerami A i B. Załóż, że wszystkie routery znają topologię całej tej sieci. W pewnym momencie łącze między A i B ulega awarii, o czym A i B od razu się dowiadują. Zalewają one sieć odpowiednią aktualizacją. Pokaż, że w okresie propagowania tej aktualizacji (kiedy dotarła ona już do części routerów a do części nie) może powstać cykl w routingu.

Rozpatrzmy sytuację jak na obrazku poniżej.



Połączenie między A i B zostało zerwane. Informacja o tym zdążyła dotrzeć już do D, ale jeszcze nie dotarła do C.

Teraz, jeśli C chce się dostać do A (z jakiegokolwiek powodu, np. A to google, a C użytkownik googla), a w jego starej (dla niego dalej aktualnej) tablicy routingu droga do A wiedzie przez B to wówczas pakiet zostanie wysłany do B. Następnie pakiet ten B wyśle dalej z powrotem do C (jedyna ścieżka w grafie). C otrzyma swój pakiet i ponownie wyśle go swoją optymalną ścieżką do B.

Dopóki do C nie dotrze informacja o awarii między A i B będziemy mieli cykl $B - C$ i pakiet nie dostanie się nigdy do A.

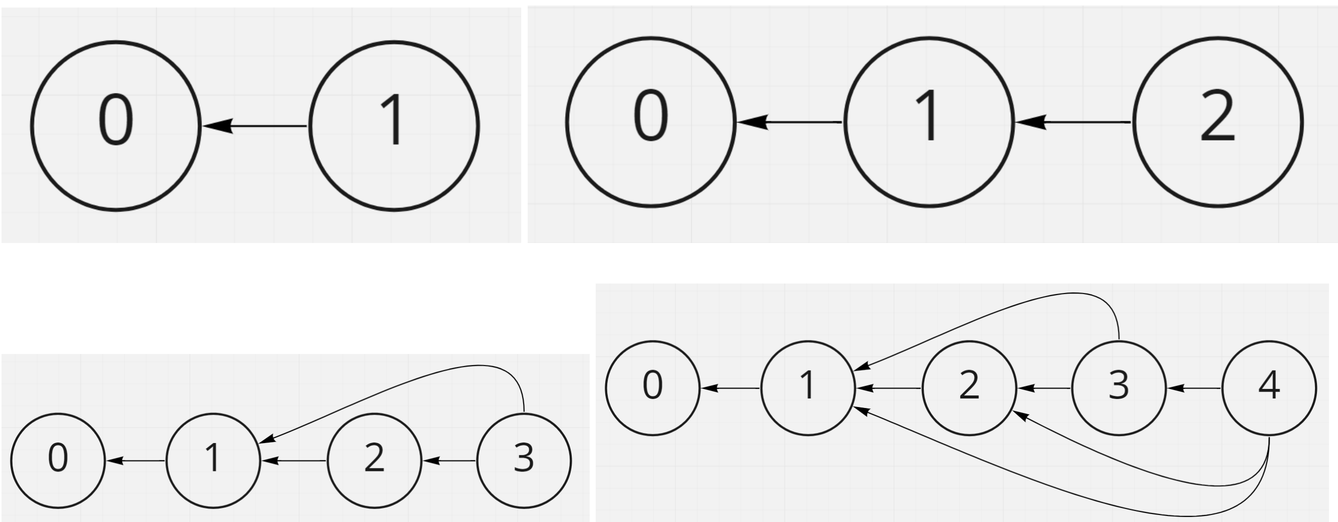
10. Zadanie

10. Załóżmy, że sieć składa się z łączy jednokierunkowych (tj. topologia sieci jest grafem skierowanym) i nie zawiera cykli. Rozważmy niekontrolowany algorytm „zalewający” sieć jakimś komunikatem: komunikat zostaje wysłany początkowo przez pewien router; każdy router, który dostanie dany komunikat przesyła go dalej wszystkimi wychodzącymi z niego krawędziami. Pokaż, że istnieją takie sieci z n routerami, w których przesyłanie informacji zakończy się po czasie $2^{\Omega(n)}$. Zakładamy, że przez jedno łącze można przesłać tylko jeden komunikat naraz, a przesłanie go trwa jednostkę czasu.

Rozważmy grafy następującej postaci:

- Graf n-wierzchołkowy o wierzchołkach $1 \ldots n$
- Dla każdego wierzchołka i, j: Jeśli $i < j$ to $j- > i$
- Dokładamy pomocniczy wierzchołek 0 oraz pomocniczą krawędź $1- > 0$

Przykład:



Komunikat będzie wysyłany przez ostatni wierzchołek z prawej. Będziemy badać ilość przesyłanych pakietów na pomocniczym połączeniu $1- > 0$. Widzimy, że:

- $T(1) = 1$ pakiet musi przejść z 1 do 0
- $T(2) = T(1)$ pakiet musi przejść z 2 do 1 i z 1 do 0
- $T(3) = T(1) + T(2)$ pakiet zostaje wysłany do wierzchołków $1, 2$
- $T(4) = T(1) + T(2) + T(3)$ podobnie jak wyżej
- $T(n) = \sum_{i=1}^{n-1} T(i)$

Chcemy teraz udowodnić, że faktycznie przesłanie informacji skończy się po $2^{\Omega(n)}$ krokach. Zacznijmy od spostrzeżenia, że przesyłanie informacji nie skończy się dopóki wszystkie pakiety nie przejdą przez pomocniczą krawędź $1- > 0$. Przez tą krawędź może przechodzić co najwyżej jeden pakiet jednocześnie.

Oznacza to, że nasze $T(n)$ jest dolnym ograniczeniem czasu przesyłania informacji. Pokażemy teraz, że istotnie spełnia wymóg zadania.

Teza:

$T(n) = 2^{n-2}$ dla $n >= 2$

Dowód indukcyjny:

Baza indukcji:

$T(2) = 1 = 2^{2-2} = 2^0$

Krok indukcyjny:

Załóżmy, że teza zachodzi dla dowolnego k mniejszego od n. Rozważmy $T(n)$. Z definicji:

$$T(n) = \sum_{i=1}^{n-1} T(i) = \sum_{i=1}^{n-2} T(i) + T(n-1) = T(n-1) + T(n-1) = 2 \cdot T(n-1)$$

Z założenia indukcyjnego:

$$2 \cdot T(n-1) = 2 \cdot 2^{n-3} = 2^{n-2} = T(n)$$

Co kończy nam dowód indukcyjny.

Teraz wiemy, że czas przesyłania informacji jest ograniczony z dołu przez 2^{n-2}. Chcemy, żeby przesyłanie trwało $2^{\Omega(n)}$.

Zauważmy, że $n - 2 = \Omega(n)$, bo $\lim_{n\to\infty} \frac{n-2}{n} > 0$ (granicowa definicja $f(n) = \Omega(g(n))$)

Zatem czas przesyłania informacji w naszych grafach jest równy $2^{\Omega(n)}$ - co należało dowieść.