HÁLÓZATI RENDSZEREK
ÉS SZOLGÁLTATÁSOK
TANSZÉK

HÁLÓZATOK ALAPJAI ÉS ÜZEMELTETÉSE

Útvonalválasztás

2023. április 21.

Mészáros András

BME Hálózati Rendszerek és Szolgáltatások Tanszék

meszarosa@hit.bme.hu

MŰEGYETEM 1782

MA MIRŐL LESZ SZÓ?

1. Irányítási tábla
2. ICMP
3. Statikus irányítás
4. Dinamikus irányítás

A fóliák elkészítéséhez felhasználtuk Jim Kurose és Keith Ross „Számítógép hálózatok működése” című könyvéhez készült fóliákat

ROUTING TÁBLA

- Címzés, továbbítás, útvonalválasztás
- Irányítási, továbbítási szabályok listája
  - Lokálisan érvényes
  - Célszerű globális ismeret alapján (is) kitölteni
- A szabályok bejegyzések
  - (Al)hálózatokra vonatkoznak
  - A CIDR elveit figyelembe véve, a maszkot is tartalmazzák
  - A következő lépést mutatják
    - Kimenő interfészt megadva
    - Hálózatot megadva – rekurzívan („Arra van, mint a …”)
  - Időről időre változhatnak
  - Átfedhetnek

Cél	Irány
10.9.42.64 / 26	GigabitEthernet 0/3
192.168.3.0 / 24	GigabitEthernet 0/1
10.9.32.0 / 19	192.168.3.1

TOVÁBBÍTÁS A TÁBLA ALAPJÁN

- Több bejegyzésre is illeszkedő cím
  - Gyakori
  - A leghosszabban illeszkedőt használjuk – **Longest Prefix Matching**
  - Prefix – a maszkolással adódó hálózati cím
  - Prefix hossz – maszk hossza
- A CIDR miatt van lehetőség címek összefogására
  - Nagyobb tartományok egy bejegyzésként
  - Kivételek külön bejegyzésként
- Nulla hosszú maszk
  - Mindenre illeszkedő tartomány
  - Egy bit sem számít a címből
  - 0.0.0.0 / 0
  - **Alapértelmezett út, default route**
  - Bármilyen másik illeszkedés hosszabb prefixet jelent

LONGEST PREFIX MATCHING

- Példa irányítási tábla

Cél	Irány
10.9.42.64 / 26	GigabitEthernet 0/3
192.168.3.0 / 24	GigabitEthernet 0/1
10.9.32.0 / 19	192.168.3.1

- Továbbítandó datagram célcíme
  - 192.168.3.78

    A 24 bites maszk miatt illeszkedik a 2. bejegyzésre – Gi0/1
  - 10.9.55.74

    0000 1010.0000 1001.0011 0111.0100 1100

    a 3. bejegyzésre illeszkedik – Gi0/1
  - 10.9.42.74

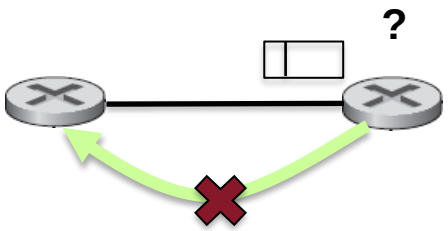
    0000 1010.0000 1001.0010 1010.0100 1100

    az 1. bejegyzésre illeszkedik

    a 3. bejegyzésre illeszkedik

PROBLÉMA A TOVÁBBÍTÁSNÁL

- Nincs illeszkedő bejegyzés
  - Küldjük vissza?
  - **Dobjuk el!**
  - Jelezzünk vissza
- A visszaküldés nagyon nem jó ötlet
  - Lokális és egyszerű döntés a továbbítás
    - Nem függ attól, hogy merről jött a datagram
  - Hurkot alakítanánk ki
    - Az előző router újra csak nekünk küldené
- **Hurok**
  - **Kerülendő**
  - Gondos konfigurációval elkerülhető
  - Feleslegesen keringő datagramok, felesleges terhelés
  - A lokális és egyszerű döntések miatt a tábla alapján nem ismerhető fel
  - Kezelése a datagram TTL mezőjének figyelésével



MA MIRŐL LESZ SZÓ?

1. Irányítási tábla
2. ICMP
3. Statikus irányítás
4. Dinamikus irányítás

ICMP

- Hogyan jelezzünk vissza?
  - A hálózati réteghez tartozó vezérlési protokollal
  - **Internet Control Message Protocol**
- A hosztok és routerek használják hálózati információk terjesztésére
  - **Hibajelzések**: elérhetetlen hálózat, hoszt, port, protokoll
  - **Echo-kérés és -válasz**: a ping ezen alapul
- IP datagramokba csomagolt üzenetek
- ICMP üzenet: típus, kód, és az üzenetet kiváltó IP datagram fejléce és első 8 adatbájtja

Type	Code	Leírás
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

ICMP ÉS TRACEROUTE

- Feladat
  - Összegyűjteni a cél felé útba eső routerek listáját
  - Közben információt gyűjteni a routerek időbeli távolságáról (RTT mérése)
- Megoldás
  - A traceroute program egy UDP csomagot küld a cél címére, de a **TTL-mezőt 1-re** állítja
  - Az első routerben a TTL mező 0-ra csökken
    - el kell dobni a csomagot
    - hibaüzenetet kell visszaküldeni: **ICMP Time-Exceeded**
  - Az adatok feldolgozása után ugyanezt kell folytatni a **TTL mezőt 2-re** állítva, és így tovább…
  - Végül egy ICMP Port-Unreachable üzenetet kell kapnunk a célhoszttól
    - Olyan UDP portot használ, ami valószínűleg nincs nyitva
- Problémák
  - Már volt róla szó: dinamikus útvonalak, becsapós RTT-k
  - Az ICMP üzenetben lévő cím a routeren a bejövő interfész címe

MA MIRŐL LESZ SZÓ?

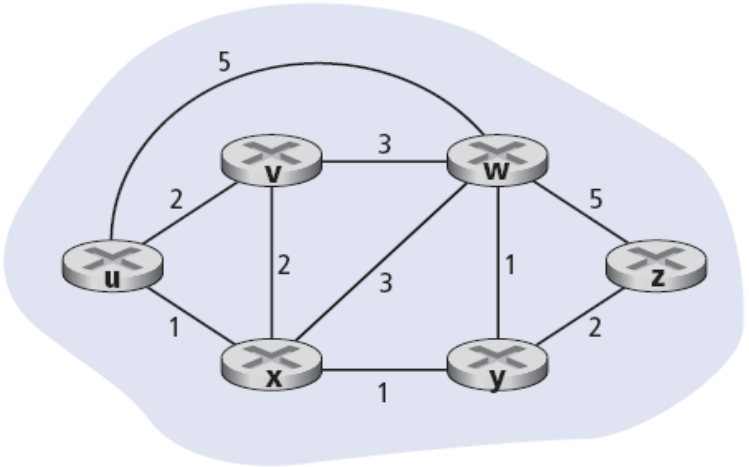
1. Irányítási tábla
2. ICMP
3. Statikus irányítás
4. Dinamikus irányítás

A HÁLÓZAT MINT GRÁF

- Tipikus interpretáció
- Egyszerűsített modell
- Gráf: G = (N,E)

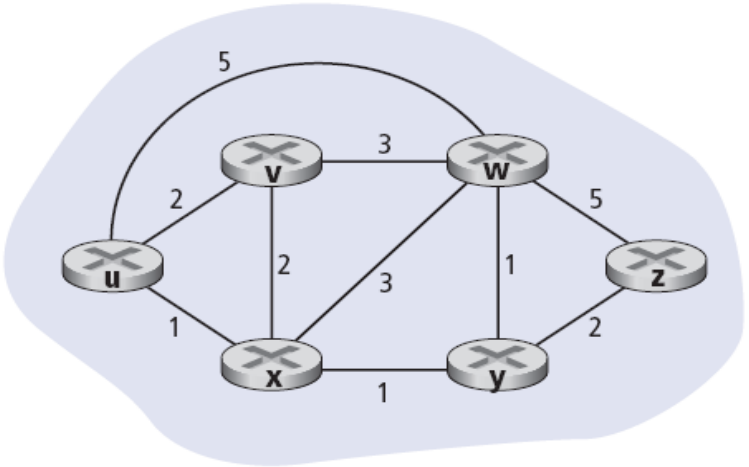
  N = **csomópontok** (routerek) halmaza =
  { u, v, w, x, y, z }

  E = **élek** (linkek, routereket összekötő alhálózatok) halmaza =
  { (u,v), (u,x), (u,w), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }
- Megjegyzések:
  - Irányított élek kellhetnek
  - Két csomópontot több él is összeköthet, de most feltesszük, hogy nincs ilyen
  - A gráfokkal más rétegeket is leírhatunk



KÖLTSÉGEK

- *c(a,b)* = az *a* és *b* csomópontokat összekötő él (link) költsége, metrikája
  - Például *c(w,z)* = 5
- A költség itt az útvonalválasztást befolyásolja, például:
  - Minden él költsége 1
  - A link sávszélességétől függ
  - A link aktuális terhelésétől függ
- Jellemzően additív mennyiség
  - Az (a_1, a_2, a_3,…, a_p) út teljes költsége = c(a_1,a_2) + c(a_2,a_3) + … + c(a_{p-1},a_p)
- Routing – legkisebb költségű út kiválasztása



ÚTVONALAK

- Alapvető kérdés: hány végponthoz kell eljuttatni egy adott datagramot, hány cél van?
  - Egy – unicast
  - Több – multicast
  - Összes – broadcast
  - Több alternatíva – anycast
- Egyetlen útvonal van (pl. unicast esetben)?
  - Mindig ugyanarra küldünk az adott cél felé
  - Nem egyenletesen osztjuk szét a forgalmat az erőforrásokon
  - Ha vannak alternatívák, pl. azonos költséggel, használjuk őket

ROUTING ALGORITMUSOK BESOROLÁSA

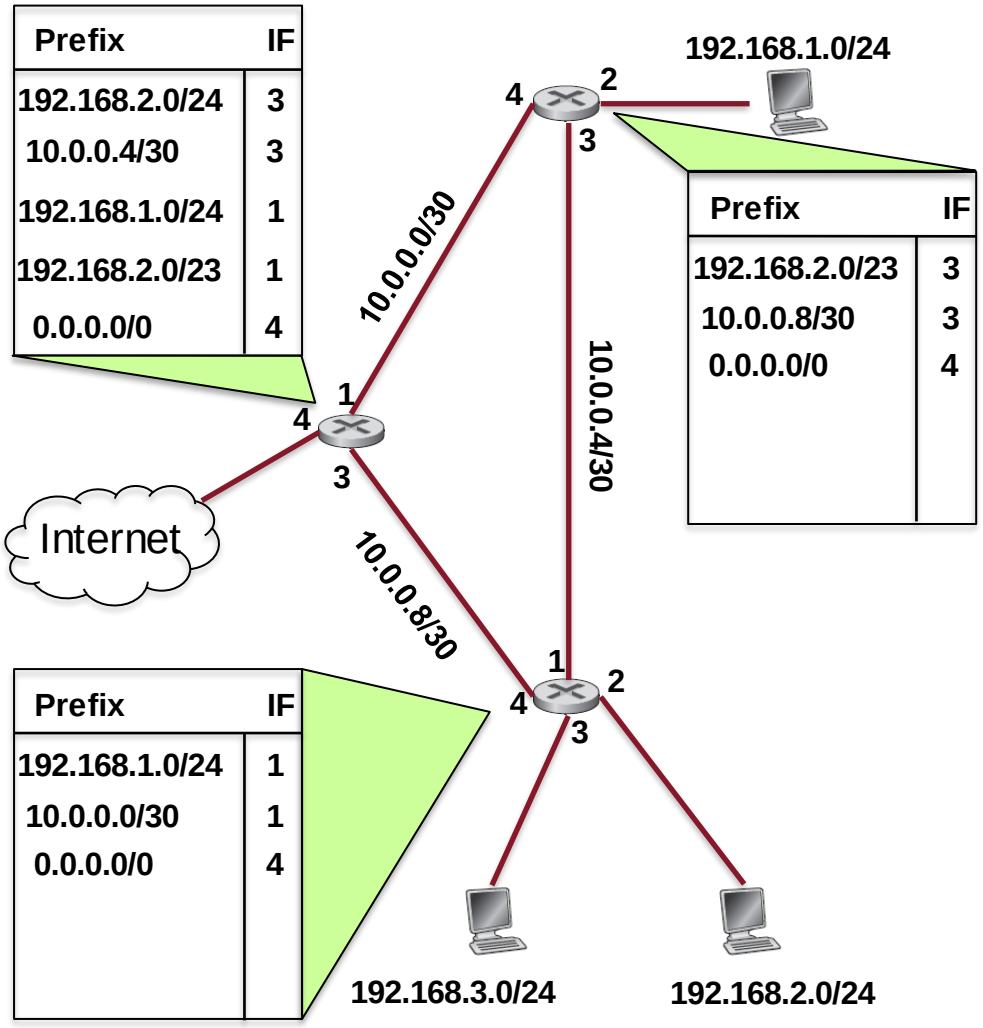
- Globális vagy elosztott információra támaszkodik?
  - **Globális információ**
    - Minden router ismeri minden részletét a gráfnak (csomópontok, élek, költségek)
    - Pl. kézi tervezés
    - Pl. link-állapot alapú protokollok
  - **Elosztott információ**
    - Minden egyes router csak a közvetlen környezetét ismeri
    - A szomszédos routerek kicserélik az ismereteiket és az alapján számolgatnak
    - Pl. távolság-vektor alapú protokollok
- Statikus vagy dinamikus?
  - **Statikus**
    - A hálózat állandóságát feltételezi
    - A routerekben ritkán frissítik a bejegyzéseket
    - Elég egyszer kiszámolni
    - Kézi konfiguráció
  - **Dinamikus**
    - A hálózat változásait követi
    - A routerek rendszeresen frissítik a bejegyzéseket
    - Rendszeresen újra kell futtatni az algoritmusokat
    - Automatizált konfiguráció
- Mikor melyik jobb?

STATIKUS BEJEGYZÉSEK

- A teljes hálózat ismeretében számítjuk ki
- Kézzel bevitt bejegyzések
  - Statikus út – static route
- Hátrányok
  - Rugalmatlan
  - Sok bejegyzés esetén nehéz elvégezni és karbantartani
- Előnyök
  - Nincs routerközi kommunikáció
    - Nem terheli a hálózatot
    - Biztonságosabb
  - Nem terheli a routert
  - Nem függ a többi routertől
- Jellegzetes alkalmazás
  - Kisebb hálózatrészekben
  - A dinamikus megoldás kiegészítéseként
  - Egyetlen linkkel bekötött hálózatrészek esetén
  - Alapértelmezett esetekre
- Mikor hat?
  - Ha a Longest Prefix Matching után választani kell, akkor a statikus utat választja
  - Több statikus út is megadható egy hálózathoz (prefixhez)
    - Különböző költség
    - Tartalék útvonal

STATIKUS ROUTING ILLUSZTRÁCIÓ

- Példa
  - Három router teljesen összekötve
  - Egyetlen kijárat az Internet felé
  - A címtervezésen már túl vagyunk
  - Nem számítunk bővítésre
- Statikus bejegyzések
  - Statikus út
  - Tartalék út
  - Alapértelmezett út



MA MIRŐL LESZ SZÓ?

1. Irányítási tábla
2. ICMP
3. Statikus irányítás
4. Dinamikus irányítás

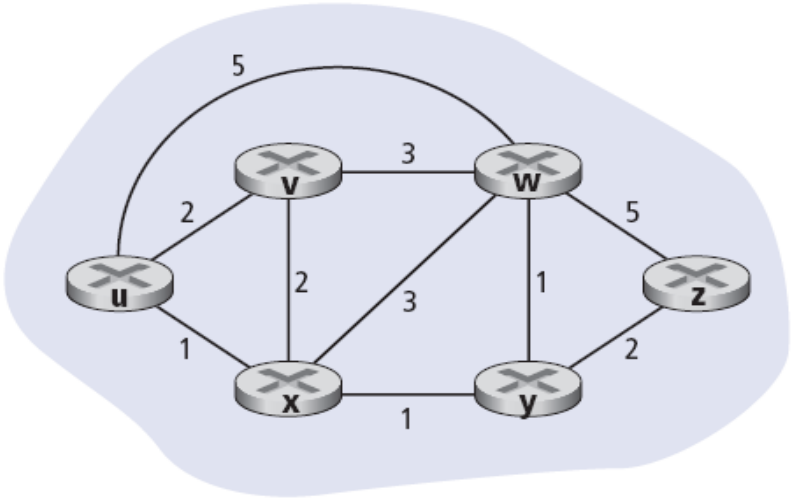
ÚTVÁLASZTÁS GLOBÁLIS ISMERET ALAPJÁN

- A teljes hálózati gráfot ismerjük
  - Minden router információkat küld a többieknek a hozzá csatlakozó alhálózatok (linkek és LANok) állapotáról
  - Mindenütt ugyanaz a kép – nagyon fontos
  - **Linkállapot (link-state) alapú**
- Optimalizált, vagy heurisztikus döntés
- Leggyakoribb megoldás: **Dijkstra algoritmusa**
  - A forrás csomópontból a többi csomópont felé menő legrövidebb utak meghatározására

- A megoldás alapötlete
  - Iteratívan bővítjük a már megoldott csomópontok halmazát
  - A halmazzal szomszédosak közül mindig legközelebbit vesszük hozzá a halmazhoz
- Jelölések
  - Az eddigi jelöléseket kiegészítve
  - A c(a,*b*) költséget ∞ -re állítjuk, ha nincs él *a* és *b* között
  - N': a már megoldott csomópontok halmaza
  - D(*v*): az eddig ismert legrövidebb távolság a forrás és *v* között az N' halmazon át, kezdetben ∞ is lehet
  - p(*v*): a forrásból *v*-be tartó eddig ismert legrövidebb úton a *v*-t megelőző csomópont

A DIJKSTRA ALGORITMUS ILLUSZTRÁCIÓJA

Iteráció	N'	D(*v*), p(*v*)	D(*w*), p(w)	D(*x*), p(*x*)	D(*y*), p(*y*)	D(*z*), p(*z*)
0	*u*	2, *u*	5, *u*	1, *u*	∞, -	∞,-
1	*ux*	2, *u*	4, *x*	1, *u*	2, *x*	∞,-
2	*uxy*	2, *u*	3, *y*	1, *u*	2, *x*	4, *y*
3	*uxyv*	2, *u*	3, *y*	1, *u*	2, *x*	4, *y*
4	*uxyvw*	2, *u*	3, *y*	1, *u*	2, *x*	4, *y*
5	*uxyvwz*	2, *u*	3, *y*	1, *u*	2, *x*	4, *y*

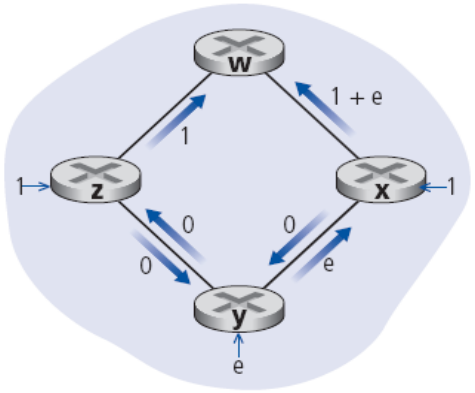


DIJKSTRA ALGORITMUS JELLEMZŐI

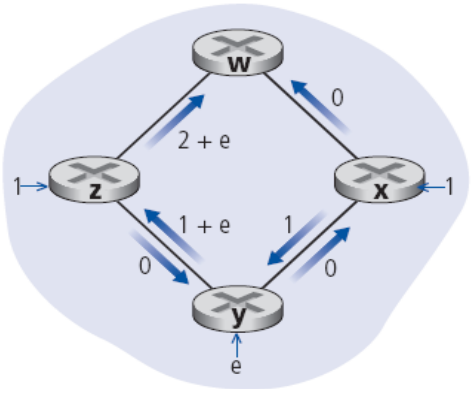
- Az algoritmus futási ideje
  - **Számítási komplexitás** n csomópont esetén
  - Minden iterációban meg kell vizsgálni az N' halmazon kívüli elemeket
  - Összesen n(n+1)/2 lépés, $O(n^2)$ komplexitás
  - Gyorsítható az adatok trükkös tárolásával: O(n log n)
- A csomópontok ugyanazt futtatják, csak más forrással
- Mikor kell újra futtatni?
  - Minden költségváltozást le kell követni **újraszámítással**
  - Nem mindig van szükség az összes út (a teljes fa) újraszámítására

OSZCILLÁCIÓ

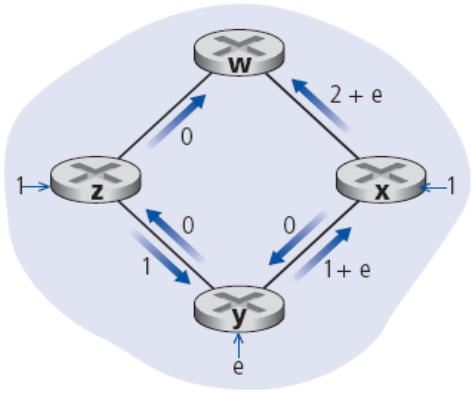
- Bizonyos esetekben gyakori újraszámítás kellhet
  - Az útvonalak ide-oda váltogatnak
  - Például terheléstől függő metrika esetén
- További zavarok okai
  - Az egyes csomópontok függetlenül (nem szinkronizálva) számolnak
  - A linkállapot információs csomag nem mindenkihez jut el
  - Egy link két irányában eltérő metrika



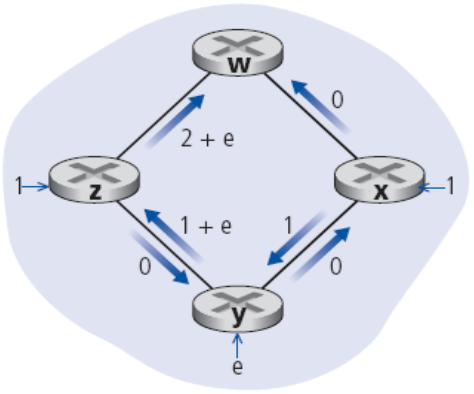
1. Kezdeti routing



2. *x* és *y* átáll a *w*-be *z*-n keresztül vezető útvonalakra



3. *x*, *y* és *z* átáll a *w*-be *x*-n keresztül vezető útvonalakra



4. *x*, *y* és *z* átáll a *w*-be *z*-n keresztül vezető útvonalakra

ÚTVÁLASZTÁS ELOSZTOTT ISMERET ALAPJÁN

- A hálózati gráfnak csak egy részét ismerik a csomópontok
  - Minden router csak a szomszédjainak küld információt arról, hogy tőle milyen messze vannak az alhálózatok
  - Sehol sincs teljes hálózati kép
  - **Távolságvektor (distance-vector) alapú**
- Optimalizált döntés, de elég hiányos információ alapján
- Megoldás: **Bellmann-Ford algoritmus**
  - A forrás csomópontból a többi csomópont felé menő legrövidebb utak meghatározására

- Jelölések
  - Az eddigi jelöléseket kiegészítve
  - A $d_a(b)$ távolság az *a*-ban ismert legkisebb távolság *b*-hez
- A megoldás alapötlete
  - Rendszeresen frissítjük a távolságokat (B-F egyenlőség)

  $d_a(b) = \min_m \{c(a,m) + d_m(b)\}$
  - A minimum képzésben *a*-nak az összes *m* szomszédját figyelembe vesszük
- A megoldás részletei
  - Egy csomópont tárolja a saját távolságvektorát

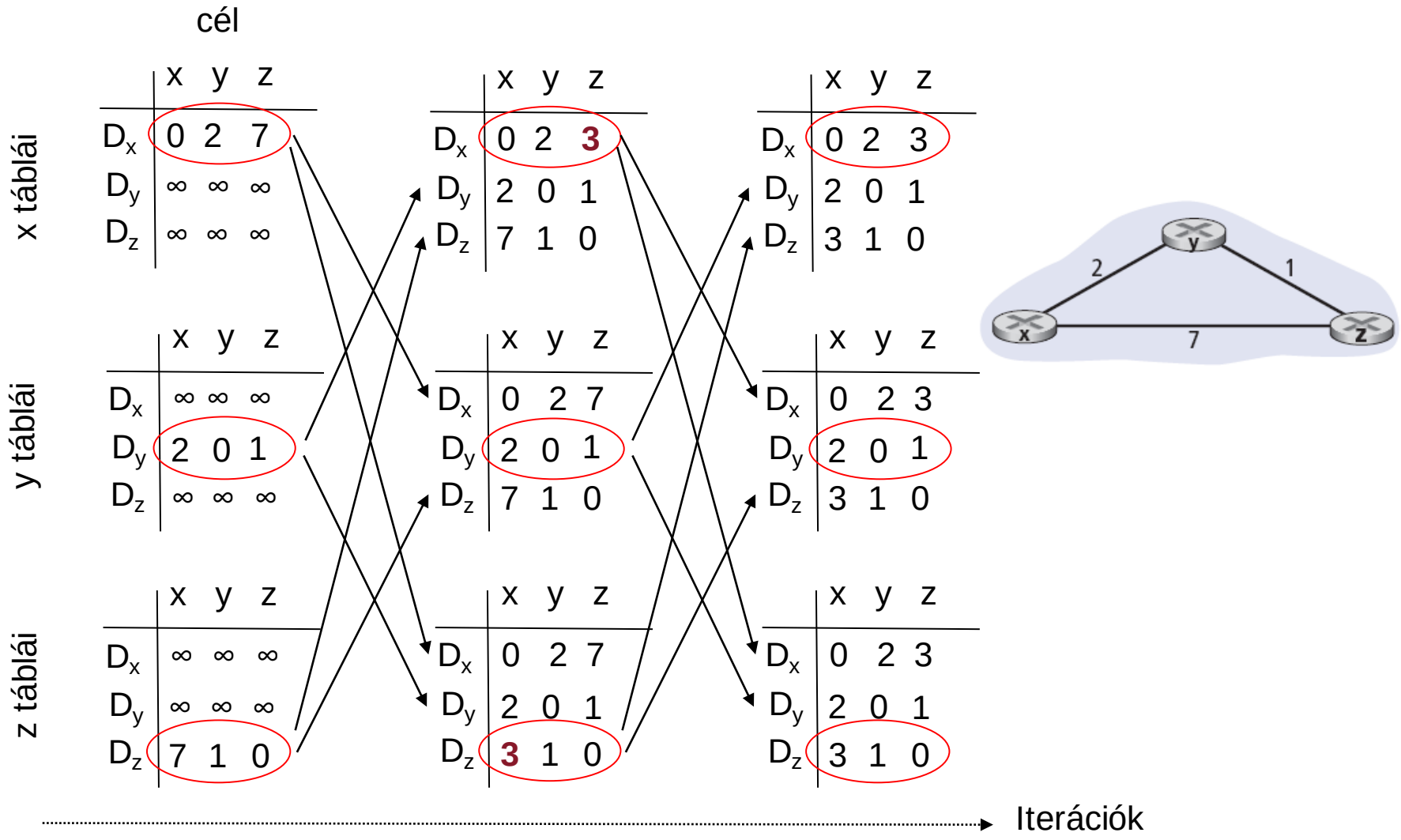
  $\mathbf{D}_a = [d_a(b): b \in N]$
  - Feldolgozza a szomszédoktól kapott távolságvektorokat

  $\mathbf{D}_m = [d_m(b): b \in N]$

TÁVOLSÁGVEKTOR ALAPÚ ROUTING

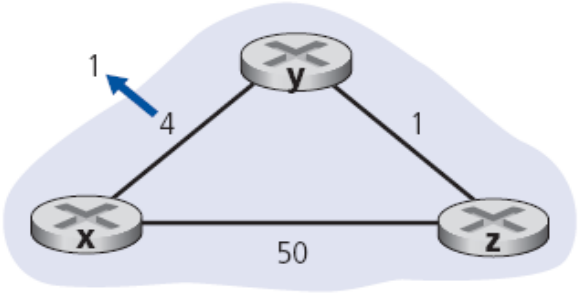
- Egy csomópont szétküldi a szomszédainak a távolságvektorát
  - Bekapcsolás után (csak a hozzákapcsolt LAN-okat tartalmazza)
  - Ha változik a vektor
- Egy csomópont frissíti a távolságvektorát
  - Amikor egy szomszédjától új vektort kap
  - Ha változnak a hozzákapcsolt LAN-ok költségei
  - Javíthatók a távolságok a B-F egyenlőség alapján
- A csomópont eltárolja minimális távolsághoz tartozó irányt
- Aszinkron számítás
- Iteratív számítás
- Elosztott, aggregált információk alapján
- Ideális esetben konvergál a minimális költségű irányokhoz

A TÁVOLSÁGVEKTOR ALAPÚ ALGORITMUS ILLUSZTRÁCIÓJA



LINK-KÖLTSÉG JAVULÁS

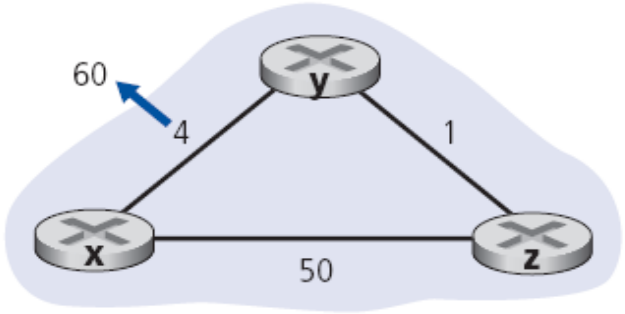
- A csomópont érzékeli, hogy egy hozzákapcsolt link költsége csökkent
- Újraszámolja a távolság vektorát
- Ha volt változás, szétküldi a szomszédoknak
  - A szomszédoknál is csak csökkenhetnek a távolságok (egyetlen iteráció alatt beáll az új érték)
  - Az összes csomópontnál csak csökkenhetnek a távolságok



- A jó hír gyorsan terjed

LINK-KÖLTSÉG ROMLÁS

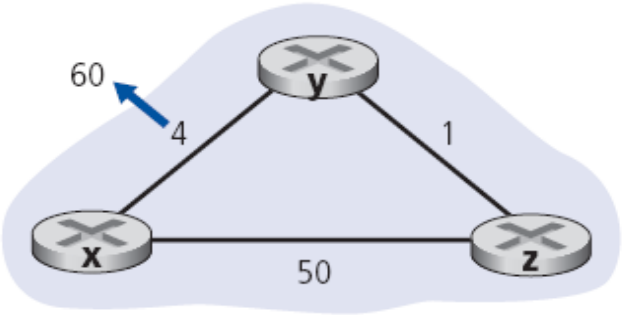
- A csomópont érzékeli, hogy egy hozzákapcsolt link költsége nőtt
- Újraszámolja a távolság vektorát
- Ha volt változás, szétküldi a szomszédoknak
  - A szomszédoknál a korábbi információk alapján lehetnek rövidebbnek tűnő utak másfelé, és ezt hirdetik is
  - Sok iteráció után derül ki, hogy tévedés volt, és addig **hurkok** is lehetnek



- A rossz hír lassan terjed
  - **Végtelenig számolás** (count-to-infinity)

VÉGTELENIG SZÁMOLÁS MEGAKADÁLYOZÁSA

- Az volt a baj, hogy
  - A z azt hirdette y felé, hogy tud egy 5 hosszú utat x felé
  - Azután y hirdette z felé, hogy ő egy 6 hosszút tud
  - És így tovább
- Ne hirdessük a célt abba az irányba, amerre irányítjuk a forgalmat
  - **Split horizon** elv
- Még jobb, ha kifejezetten jelezzük is ezt
  - Végtelen értéket hirdetünk
  - **Poisoned reverse**
- Nagyobb hálózatokban ez sem mindig elég



ÖSSZEHASONLÍTÁS

- Üzenetek száma
  - LS: n darab csomópont, E darab él: sok, O(nE) üzenet
  - DV: nagyobb üzenetek, de csak a szomszédok felé
- Konvergálás sebessége
  - LS: gyors
    - Linkállapotok szétküldésének ideje
    - Oszcillálhat
  - DV: változó
    - Ideiglenes hurkok kialakulhatnak
    - Végtelenig számolás problémája
- Robusztusság (hibatűrés): Mit eredményez, ha meghibásodik egy router?
  - LS
    - A csomópont hamis adatokat küld a kapcsolódó linkekről
    - Az útvonalak nem biztos, hogy optimálisak maradnak
  - DV
    - A router hibás távolságvektort küld (az egész útról rossz lesz az információ)
    - Az információk elterjednek az egész hálózatban, mindenki hamis távolságokat számol és hirdet
    - Katasztrofális hatású lehet, különösen, ha kis távolságokat hirdet

HÁLÓZATI RENDSZEREK
ÉS SZOLGÁLTATÁSOK
TANSZÉK