Počítačové sítě, v. 3.5

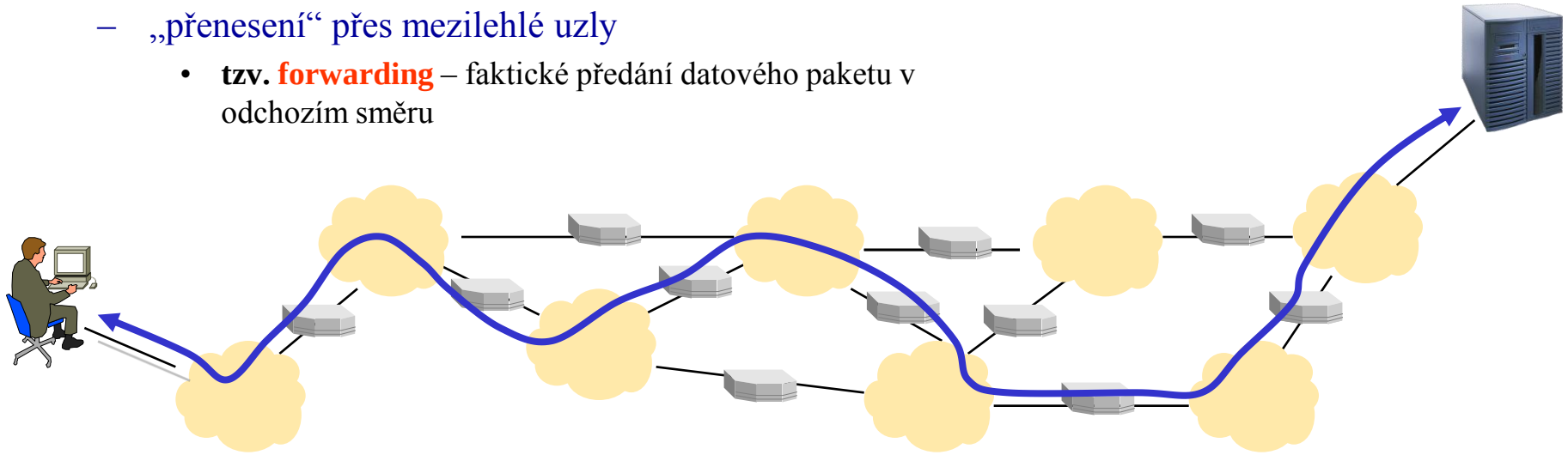
Katedra softwarového inženýrství,
Matematicko-fyzikální fakulta,
Univerzita Karlova, Praha

Lekce 9: Síťová vrstva a směrování

Jiří Peterka, 2010

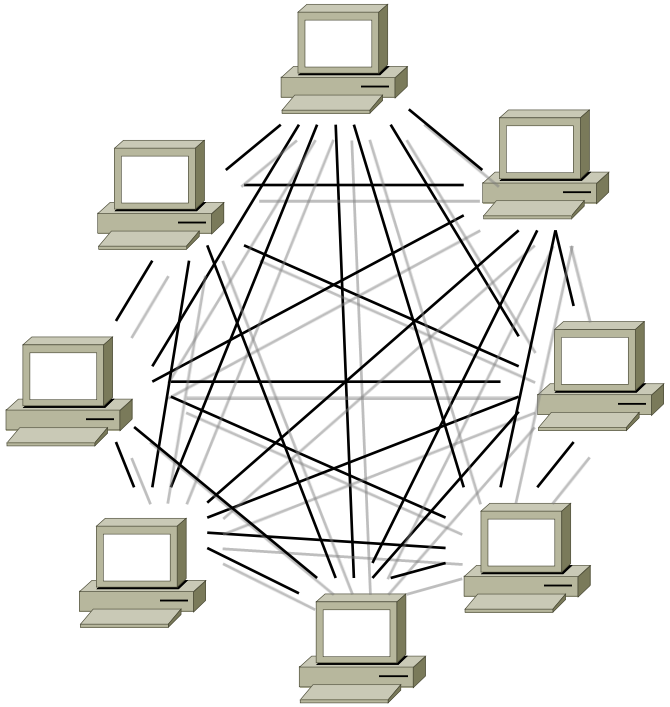
hlavní úkol síťové vrstvy

- doručovat data od jejich zdroje až k jejich koncovým adresátům
  - což typicky obnáší přenos přes mezilehlé uzly
    - linková vrstva se stará jen o doručováním k přímým sousedům (v dosahu přímého spojení) a nezabývá se přenosem "dál" ….
- k tomu je zapotřebí:
  - vyhledání vhodné (optimální) cesty k cílovému uzlu
    - tzv. **směrování** (routing), rozhodnutí o dalším (odchozím) směru přenosu
  - „přenesení“ přes mezilehlé uzly
    - **tzv. forwarding** – faktické předání datového paketu v odchozím směru
- dalším úkolem síťové vrstvy je předcházet zahlcení
  - congestion control
    - něco jiného než řízení toku
- dalším úkolem je (může být) zajištění tzv. kvality služeb
  - QoS, Quality of Service
    - mj. podpora multimediálních aplikací



požadavky na síťovou vrstvu

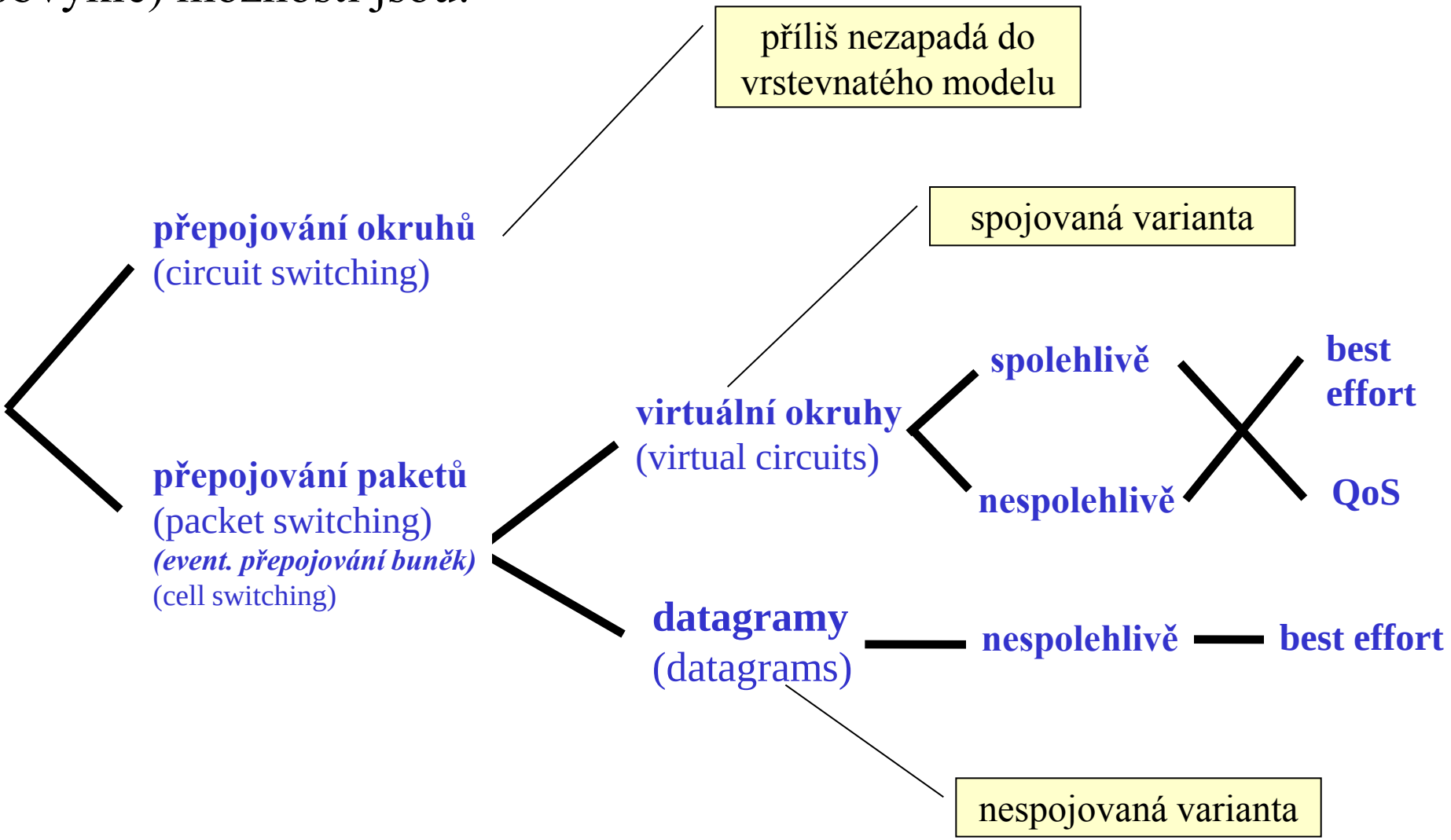
- přenosové služby, které poskytuje transportní vrstvě, by měly být:
  - funkční
    - nemělo by docházet (moc často) k zahlcení
  - nezávislé na konkrétní topologii sítě
    - aby si transportní vrstva mohla myslet, že existuje přímé spojení každého s každým
  - nezávislé na konkrétní přenosové technologii dílčích sítí
    - aby si transportní vrstva nemusela uvědomovat odlišnosti v přenosových technologiích
  - adresování by mělo být jednotné
    - v rámci LAN i WAN
      - např. IP adresy v TCP/IP



- "otevřené otázky":
  - měla by síťová vrstva fungovat spolehlivě, či naopak nespolehlivě?
  - má fungovat stylem "best effort", nebo garantovat kvalitu služeb
  - má fungovat spojovaně, nebo nespojovaně?
  - …….

možnosti fungování síťové vrstvy

(obvyklé) možnosti jsou:



ad otevřené otázky

- má síťová vrstva fungovat spojovaně, nebo nespojovaně?
- má zajišťovat spolehlivost, nebo nemusí?
  - má se soustředit jen na přenos, a nezabývat se dalšími věcmi?
  - má být jednoduchá, nebo naopak složitá (vybavená funkcemi a schopnostmi)?
- otázka ve skutečnosti zní takto:
  - kam má být umístěna složitost (inteligence)? Do síťové vrstvy, nebo do transportní?

= spojovaná, spolehlivá, QoS …

= nespojovaná, nespolehlivá, best effort …

- "*lidé od spojů*":
  - síťová vrstva by měla být spíše bohatší ....
    - měla by zajišťovat spolehlivost přenosu
    - měla by fungovat spojovaně
    - měla by ev. nabízet i další služby
      - garanci kvality
      - možnost rezervace přenosové kapacity
  - znamená to, že přenosová část sítě bude muset mít velkou vlastní inteligenci!
    - bude fungovat méně efektivně
    - někdo to bude muset zaplatit
- "*lidé od počítačů*"
  - síťová vrstva by měla být co nejjednodušší a nejvýkonnější
    - měla by se soustřeďovat jen na svůj "core byznys" – na přenos paketů
  - inteligence má být až v koncových uzlech

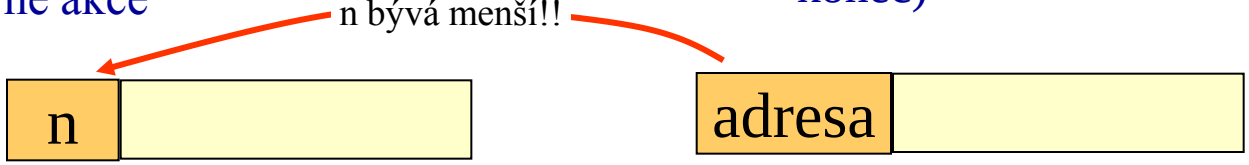
virtuální okruhy vs. datagramy

virtuální okruhy:

- přenáší se pakety
- paket je opatřen *identifikátorem virtuálního okruhu*
  - který může být i poměrně malý
- pakety cestují vždy stejnou cestou
  - je zaručeno správné pořadí doručování paketů
- mechanismus přenosu po virtuálních okruzích je stavový
  - navázáním spojení dochází ke změně stavu
  - je nutné explicitně ukončovat spojení
  - při výpadku je nutné podnikat nápravné akce

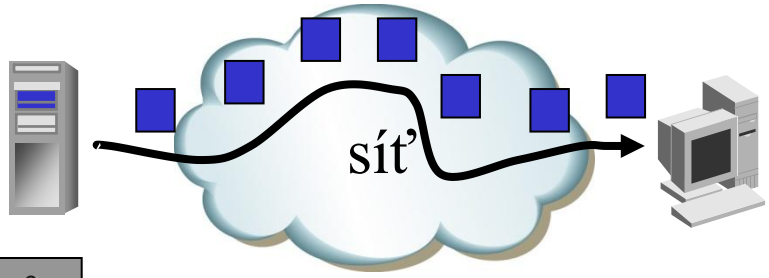
datagramová služba:

- přenáší se datagramy
- datagram je opatřen *celou adresou svého příjemce*
  - která může být dost velká
- datagramy nemusí ke svému cíli cestovat vždy stejnou cestou
  - není zaručeno pořadí doručování datagramů
- přenos datagramů je bezestavový
  - není navazováno spojení, nemění se stav
  - není nutné jakkoli ukončovat spojení (někomu signalizovat konec)

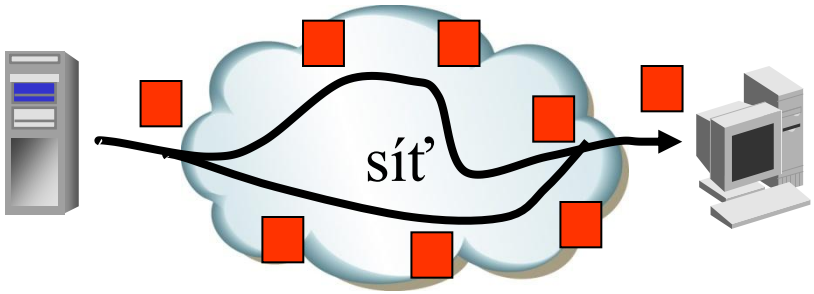


virtuální okruhy vs. datagramy

- virtuální okruhy:
  - je nutné nejprve navázat spojení
    - v rámci toho dojde k vytyčení přenosové trasy
    - routing (rozhodnutí o volbě směru) se provádí **jednou**, při navazování spojení
  - data se přenáší vždy **stejnou cestou**
    - forwarding ("předání dál") se realizuje podle předem určeného (zvoleného) směru
  - nedokáže to (samo) reagovat na dynamické změny v síti
    - výhodné pro (intenzivnější) přenosy (větších objemů dat)

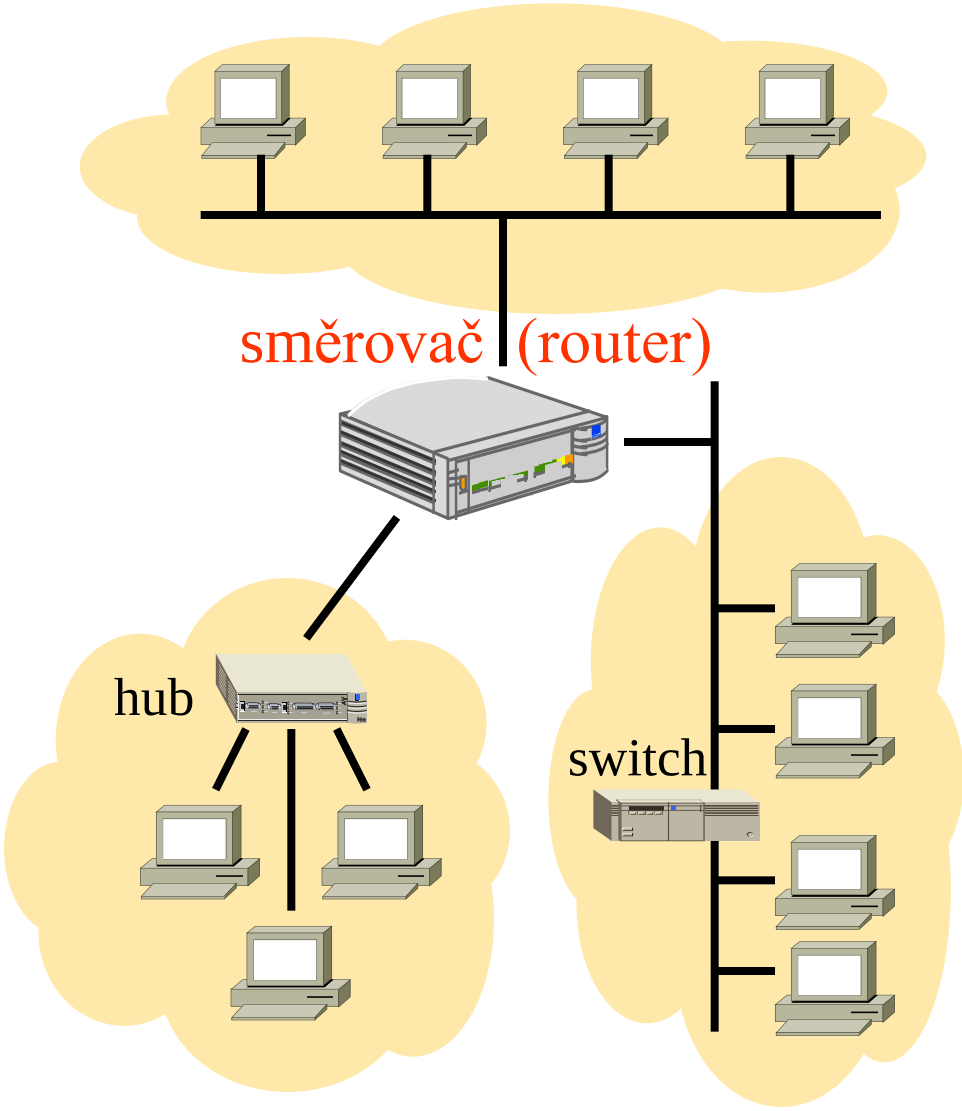


- datagramy
  - spojení se nenavazuje
    - routing (rozhodování o volbě směru) se provádí **pokaždé znovu**
      - pro každý datagram, v každém uzlu
  - data se **nemusí přenášet vždy stejnou cestou**
  - přenos dokáže reagovat na průběžné změny v síti
    - výhodné pro menší a „řidší“ (více příležitostné) přenosy



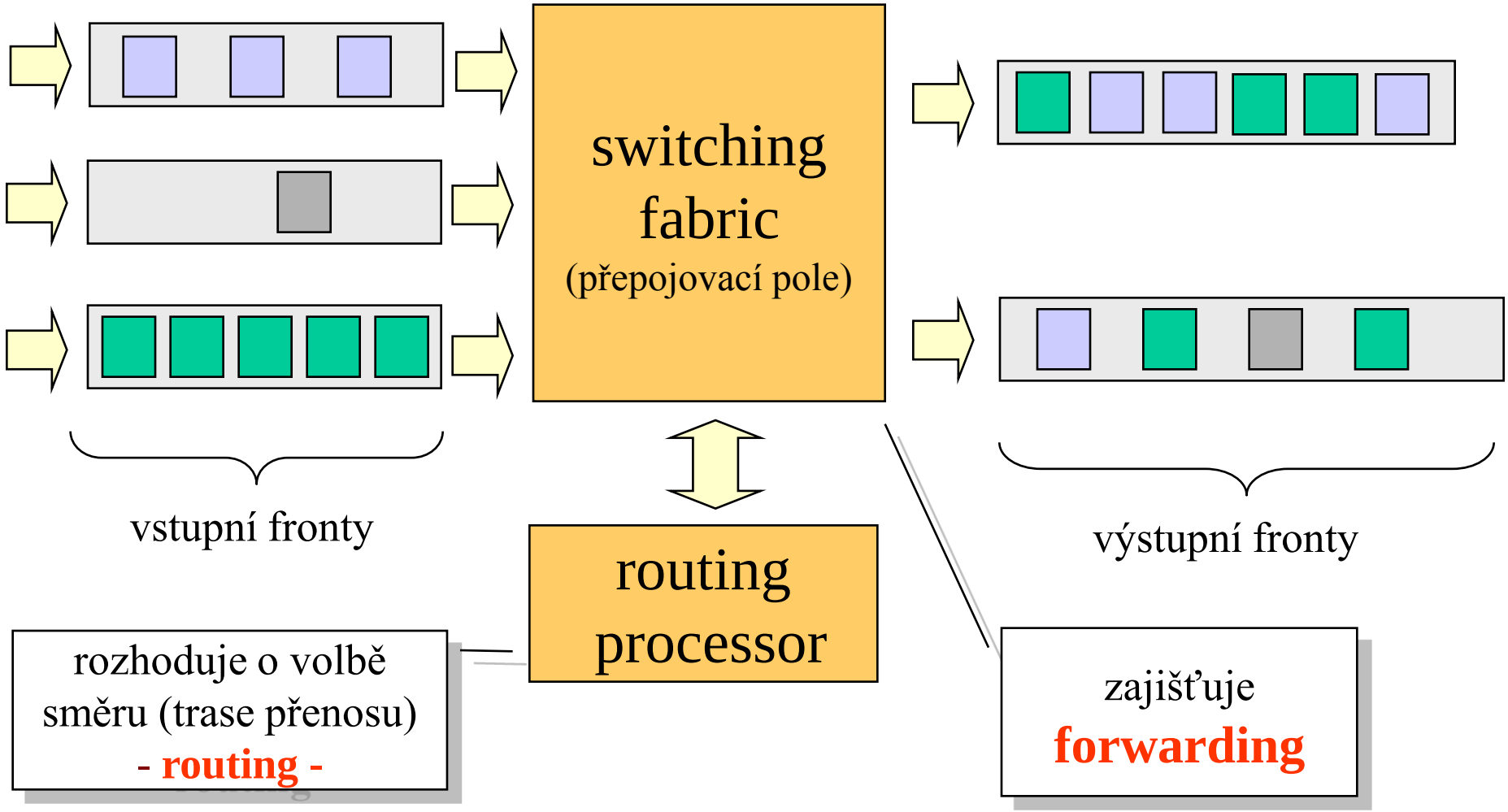
směrovač (router)

- zařízení, které zajišťuje propojení na úrovni síťové vrstvy se označuje jako **směrovač** (router)
  - tj. funguje na úrovni síťové vrstvy
  - v prostředí s přepojováním paketů
    - zajišťuje manipulaci s pakety
    - řeší vlastní volbu směru (směrování) i "realizaci přeskoku" (forwarding)
  - propojuje dvě či více sítí
    - síť = soustava uzlů vzájemně propojených na úrovni linkové vrstvy



přepínač (switch)
je zařízení, které propojuje na linkové vrstvě
opakovač (repeater)
propojuje na fyzické vrstvě

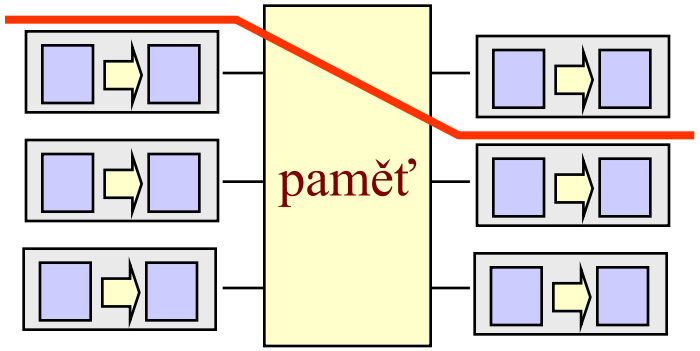
princip fungování směrovače



realizace přepojovacího pole

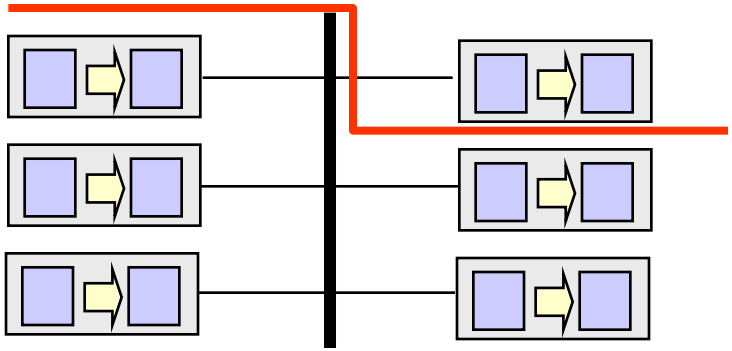
paměť (s více porty)

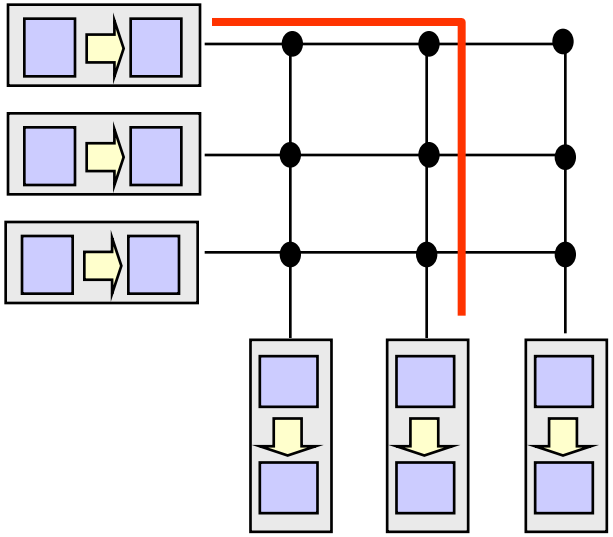
pakety se předávají mezi
frontami přes sdílenou paměť



sběrnice

fronty jsou propojeny přes sběrnici,
vždy se po ní přenáší jen jeden paket



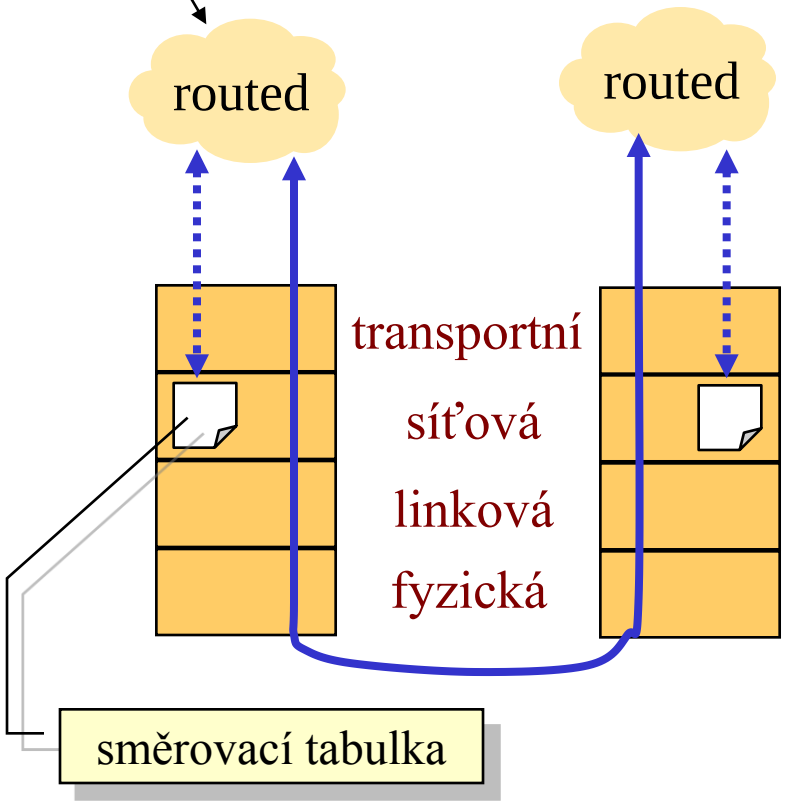


"crossbar"
propojovací síť
s N + N sběrnicemi

směrovací tabulky (routing tables)

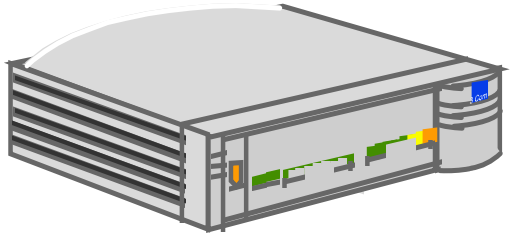
- každý směrovač musí mít určité informace o topologii sítě
  - kromě "zvláštních" metod směrování které pracují bez znalosti topologie sítě,
    - například: záplavové směrování, metoda horké brambory
- informace o topologii si směrovač udržuje v rámci své směrovací tabulky
  - u adaptivních algoritmů směrování (které se dynamicky adaptují na momentální stav sítě), musí být směrovací tabulky průběžně (dynamicky) aktualizovány
    - vhodnou výměnou aktualizačních informací mezi směrovači
  - u neadaptivních algoritmů lze tabulky naplnit jednorázově (dopředu – staticky)

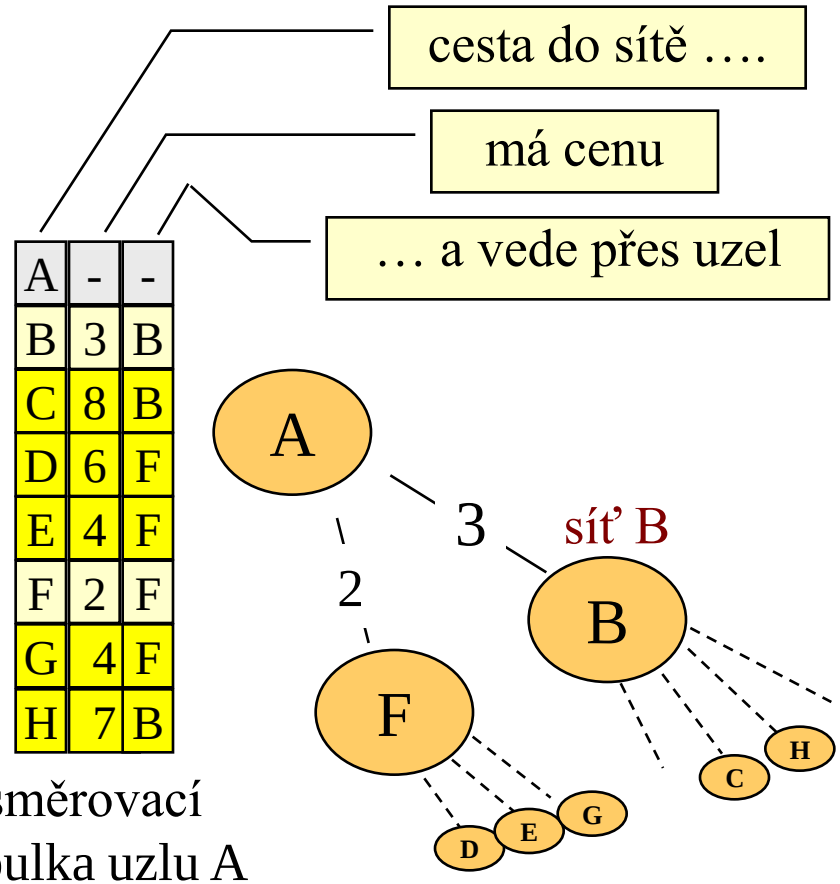
příklad: aktualizační informace zpracovává (aplikační) proces (route demon), který také upravuje směrovací tabulku (umístěnou na síťové vrstvě)



představa využití směrovacích tabulek (pro nespojovanou variantu směrování)

- směrovací tabulka obsahuje trojice údajů <cílová síť; vzdálenost; odchozí směr>
  - směrovače typicky směrují podle příslušnosti k cílové síti
- postup při zpracování paketu
  - paket obsahuje adresu cílové sítě
  - směrovač použije cílovou adresu jako klíč pro hledání v tabulce
    - prohledá tabulku, najde položku ….
  - podle nalezené položky směrovač určí odchozí směr a předá paket tímto směrem
- adaptace na změny v síti
  - obsah směrovacích tabulek se dynamicky mění
    - v závislosti na dění v síti
- (adaptivní) směrovací algoritmy
  - neustále "přepočítávají" obsah směrovacích tabulek





klasifikace algoritmů směrování

- algoritmy směrování se snaží hledat optimální cestu
- co je optimální?
  - nejkratší
    - a v jakém smyslu? V počtu přeskoků, délce kabelu či přenosové cesty?
  - nejrychlejší
    - co do přenosového zpoždění, co do délky front?
  - nejlacinější
    - co do nákladů, poplatků?
  - …….
- obecně: zavede se určitá metrika, a tou se ohodnotí jednotlivé spoje v síti
  - algoritmy hledají optimální cestu podle této metriky
  - metrika může vyjadřovat např. počet přeskoků, celkovou dobu přenosu, nebo kombinaci …..

- ne-adaptivní algoritmus:
  - nesnaží se reagovat na průběžné změny v síti (změny ohodnocení hran)
  - dokáže „vypočítat“ optimální trasy předem
  - nepotřebuje přenášet aktualizační informace
  - při výpadcích částí sítě může způsobit nefunkčnost sítě
- adaptivní algoritmus
  - snaží se reagovat na průběžné změny v síti
  - musí „počítat“ optimální trasy průběžně
  - vyžaduje pravidelný přísun aktualizačních informací

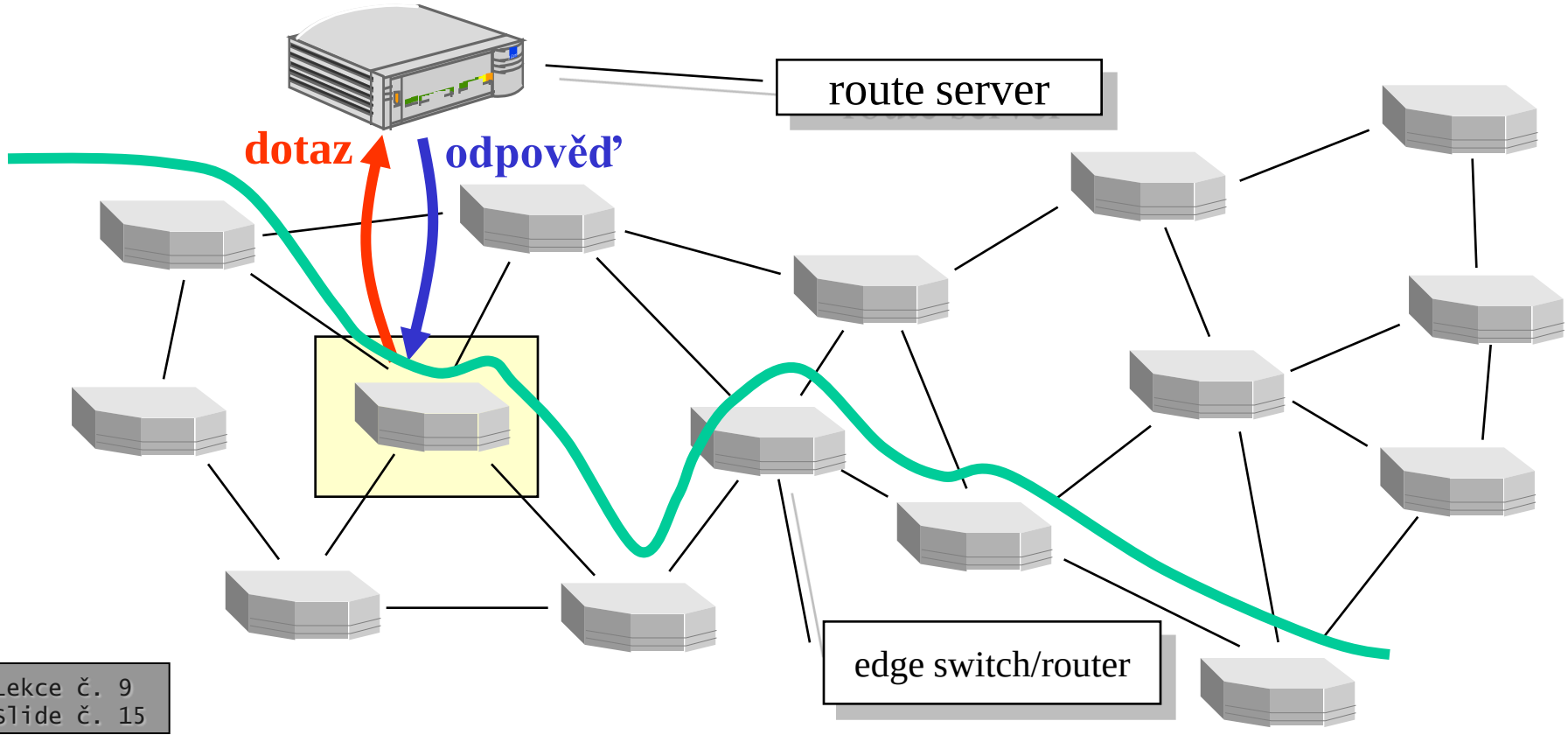
záleží také na četnosti změn v síti

klasifikace algoritmů směrování

- centralizované směrování
  - routing (rozhodování o směru) provádí jeden centralizovaný uzel
    - tzv. route server
  - jednotlivé směrovače (spíše tzv. "edge switch-e") provádí pouze forwarding
    - kdykoli neví jak, zeptají se route serveru
- izolované směrování
  - směrovače provádí routing i forwarding
  - jednotlivé uzly nespolupracují na hledání optimálních cest !!!
    - každý uzel se rozhoduje jen sám za sebe
  - příklady:
    - záplavové směrování
    - metoda horké brambory
    - metoda zpětného učení
    - source routing
    - ….
  - obecně:
    - méně efektivní, než když uzly spolupracují
- distribuované směrování
  - směrovače provádí routing i forwarding
  - uzly vzájemně spolupracují na hledání optimálních cest a na aktualizaci svých směrovacích informací
  - příklady:
    - vector distance routing
    - link state routing
- hierarchické směrování
  - obecný problém směrování:
    - směrovací informace (o topologii sítě, stavu spojů, …) jsou příliš velké, přestává to být únosné
  - řešením je dekompozice
    - soustava sítí se rozdělí na více relativně samostatných oblastí (area)
    - uvnitř oblastí se směrování řeší samostatně,
    - "mezi" oblastmi se řeší jen "na hrubo"

představa centralizovaného směrování

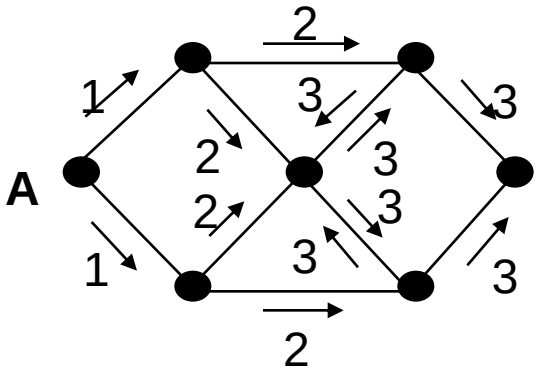
- počítá s existencí jednoho centrálního subjektu (route serveru), který:
  - sám vypočítává (jednorázově nebo průběžně) nejvýhodnější cesty
  - výsledky distribuuje všem směrovačům, které se podle nich řídí
    - a pamatují si je v cache paměti
- použitý algoritmus směrování může být adaptivní i neadaptivní
- s výpadkem route serveru přestává síť fungovat
- v praxi se moc nepoužívá
  - používají se spíše distribuované (a izolované) varianty bez centrálního prvku



záplavové směrování (flooding)

- varianta izolovaného směrování
- v každém mezilehlém uzlu je každý paket rozeslán do všech směrů které existují
  - kromě toho, ze kterého přišel
  - je to maximálně robustní
    - pokud existuje cesta k cíli, je nalezena (dokonce ta "nejkratší")
  - realizace je velmi jednoduchá
    - nevyžadují se žádné směrovací tabulky
    - není nutné přenášet žádné aktualizační informace o stavu sítě
  - problémy jsou s eliminací nadbytečných paketů
    - řeší se např. vkládáním čítačů do všech paketů, při dočítání k nule je paket eliminován
    - nebo pamatováním již prošlých paketů a eliminací duplikátů
      - podle jejich unikátního ID

- jako metoda pro směrování "běžných" dat se používá ve speciálních sítích
  - například vojenských
- častěji se používá jako mechanismus šíření "speciálních" dat
  - například aktualizačních informací
  - pro hledání cesty
  - …..
- a pro speciální účely
  - například pro aktualizaci distribuovaných databází
  - pro distribuci vyhledávacích dotazů distribuovaným vyhledávacím službám



další příklady izolovaného směrování

metoda horké brambory

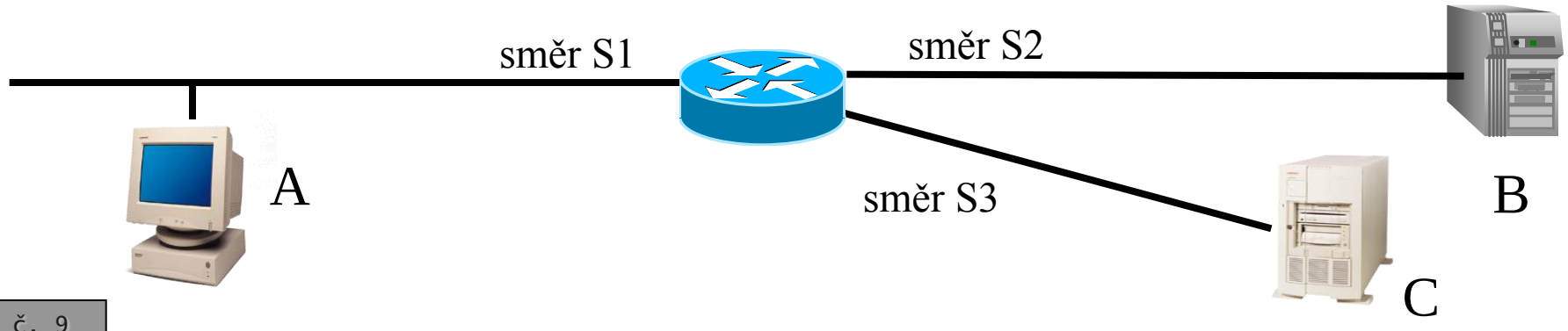
- idea: když začíná být zle, je vhodné se paketů zbavovat co možná nejrychleji
  - být zle = když začínají přetékat výstupní fronty
  - co nejrychleji se zbavit = odeslat tím směrem, který je momentálně nejméně vytížen
    - má nejkratší výstupní frontu
- používá se to jako doplňková metoda pro případ, kdy jiná (základní) metoda směrování vede k přeplnění front
  - nebo v situaci, kdy je třeba zpracovat více paketů najednou, ale nelze je všechny odeslat stejným směrem
    - odešlou se jen některé, ostatní stylem "horké brambory"

náhodné směrování

- paket je směrován do náhodně zvoleného směru
  - jako "samostatná" metoda směrování nemá příliš velký smysl
    - zvolené cesty jsou sub-optimální, nemusí vést k cíli
  - používá se pro speciální účely
    - jako doplněk k "řádné" metodě směrování
      - například při přetékání výstupních front (jako alternativa k metodě horké brambory)
      - když směrovač nestíhá řádně zpracovat všechny pakety (když jeho rozhodovací kapacita nestačí, jsou některé pakety směrovány "řádně"a jiné náhodně)

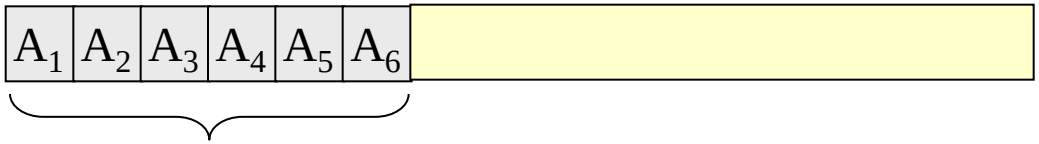
metoda zpětného učení

- každý směrovač si sám získává potřebné informace o topologii z paketů, které skrz něj prochází
  - zpětně se z nich učí znát topologii sítě
- princip (zpětného) učení:
  - na počátku směrovač nic neví a směruje záplavově
  - když přijme paket od uzlu A ze směru S1, odvodí si z toho že A leží ve směru S1
    - když dostane odpověď od uzlu B ze směru S2 (určenou uzlu A), odvodí si
      - že uzel B leží ve směru S2
      - přepošle paket cíleně ve směru S1 pro A
- zpětné učení se používá spíše na linkové vrstvě
  - u ethernetových mostů a přepínačů – pro volbu směru k nejbližším sousedům
    - pro směrování ve větších sítích není zpětné učení příliš vhodné

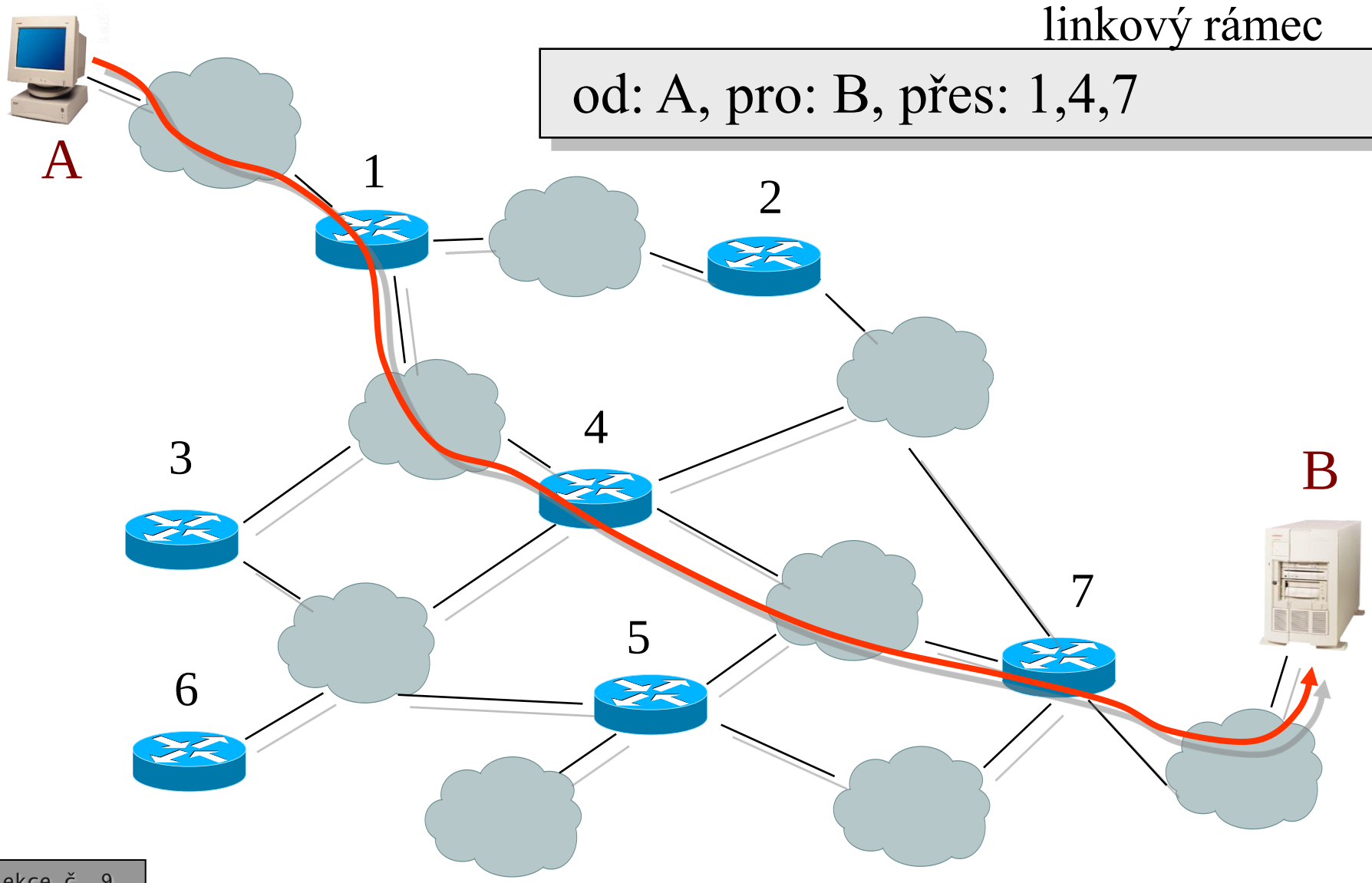


source routing

- source routing
  - doslova: *zdrojové směrování, směrování prováděné zdrojem*
- podstata:
  - každý jednotlivý rámec si v sobě nese úplný itinerář
    - úplný seznam uzlů, přes které má projít
  - tento „itinerář“ sestavuje odesílající uzel
    - proto „source“ routing
  - má to blíže k síťové vrstvě než k vrstvě linkové
    - v názvu to má „směrování“ (routing)
- source routing je technika používaná na úrovni linkové vrstvy !!!
  - ačkoli "směrování" naznačuje síťovou vrstvu
  - používá se v Token Ringu
- kde vezme odesílající uzel znalost o topologii sítě, na základě které sestaví úplný itinerář?
  - Před odesláním paketu (paketů) vyšle do sítě průzkumný paket
  - průzkumný paket (spíše rámec) se šíří záplavově (jako lavina), až dorazí ke svému cíli
  - po dosažení cíle se průzkumný paket vrací a nese v sobě údaj o cestě, kterou se k cíli dostal
- záplavové rozesílání není moc šetrné k přenosové kapacitě
  - ale najde skutečně „nejkratší“ cestu
  - není to ale příliš adaptivní
    - po počátečním nalezení cesty



představa Source Routing-u



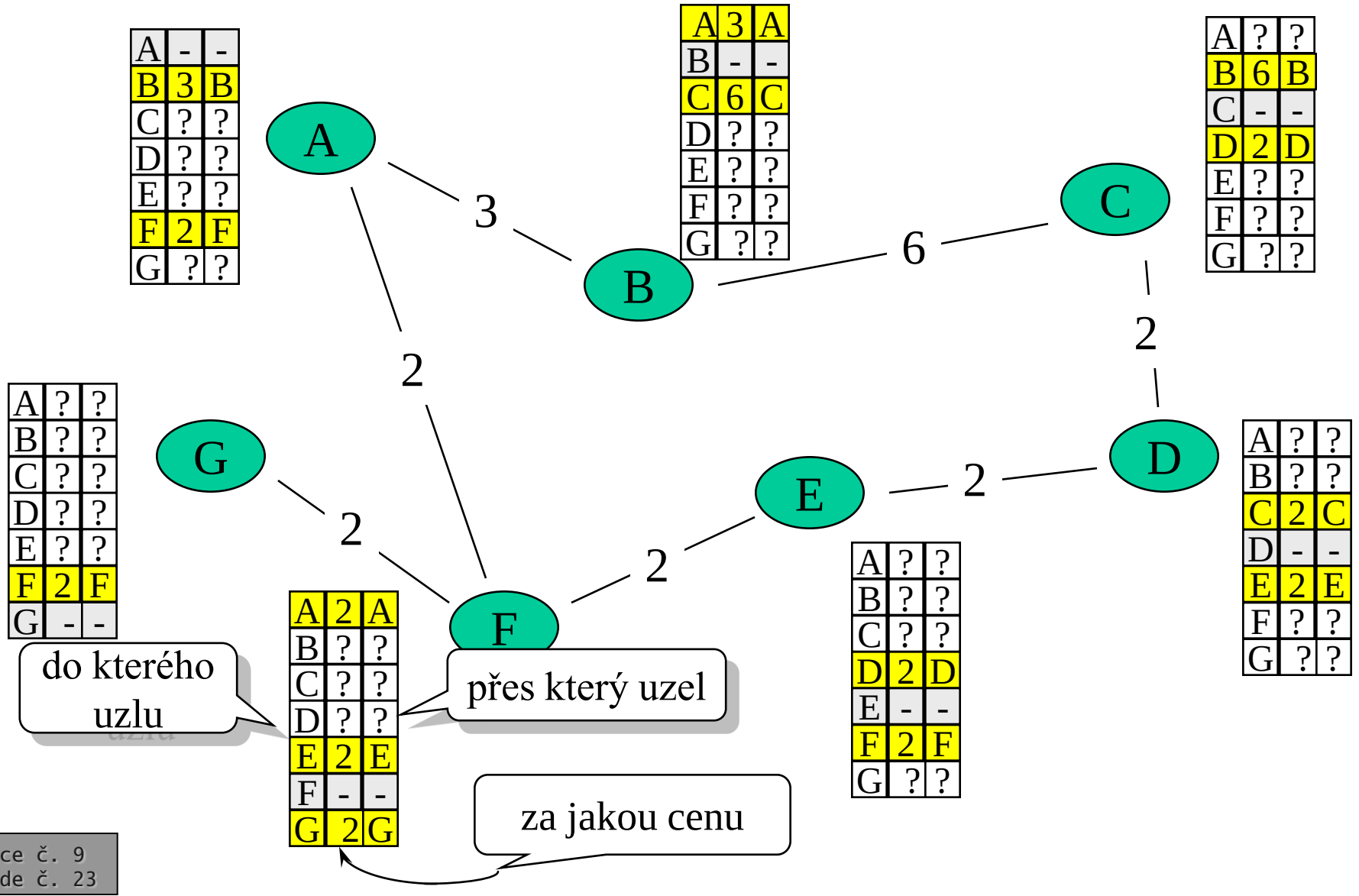
distribuované směrování

- nejčastěji používaná varianta směrování
- jednotlivé směrovače vzájemně spolupracují
  - posílají si aktualizační informace o změnách v síti
    - pokud je směrování adaptivní a reaguje na dění v síti
  - případně "části výpočtu"
- spolupráce směrovačů může být různá:
  - výpočet optimálních cest je distribuovaný
    - každý počítá kus, vzájemně si předávají části výpočtů
    - nevýhoda: když jeden udělá chybu, "splete" i ostatní
  - každý si počítá optimální cesty sám
    - uzly si posílají jen "podklady" (informace o dostupnosti a změnách)
- výpočet optimálních cest
  - je klasickou úlohou z teorie grafů
  - používají se například:
    - Bellman-Fordův algoritmus
    - algoritmus Ford-Fulkersona
- z hlediska zatížení sítě (přenosových cest) je důležité:
  - jaké objemy (aktualizačních) informací se přenáší
    - může být limitujícím faktorem
  - jak často
    - pravidelně
    - jen při změně
- základní varianty směrování:
  - vector distance routing
    - např. protokol RIP
    - dnes se hodí již jen pro menší sítě
  - link state routing
    - např. protokol OSPF
    - hodí se i pro větší sítě

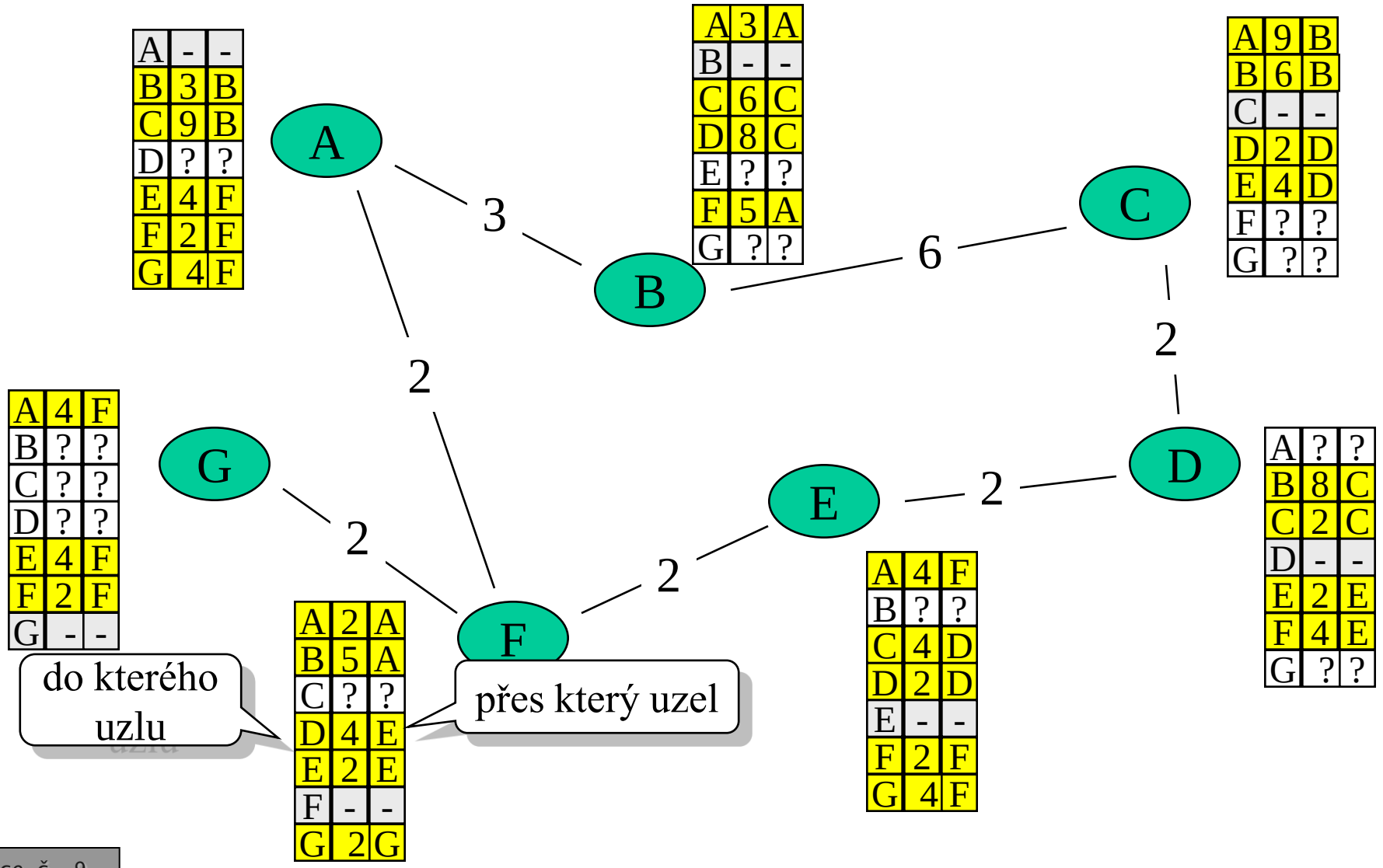
vector-distance routing

- idea:
  - každý směrovač si udržuje tabulku svých nejmenších „vzdáleností“ od všech ostatních uzlů ("vektorů")
  - směrovače si tyto informace vzájemně vyměňují
    - informace typu:
      - já se dostanu k uzlu X za cenu Y
    - jde vlastně o průběžnou výměnu obsahu celých směrovacích tabulek
  - … ale výměna probíhá jen mezi přímými sousedy, ne mezi všemi směrovači sítě !!!!!
  - všechny směrovače si průběžně vypočítávají nové nejkratší vzdálenosti
    - na základě vektorů, které dostávají od svých sousedů
    - výpočet optimálních cest je fakticky distribuovaný
      - když někdo udělá chybu, splete i ostatní
- objemy přenášených dat (pro potřeby aktualizace) jsou hodně velké
  - není to vhodné pro velké sítě
- problémy jsou i s konvergencí:
  - „dobré zprávy“ se šíří rychle
    - to, že někde existuje kratší cesta
  - „špatné zprávy“ se šíří pomalu
    - to, že někde přestala být cesta průchodná
  - problém „count-to-infinity“
    - hodnota cesty přes neprůchodnou cestu se zvyšuje v každém kroku o 1
    - trvá to hodně dlouho, než se hodnota zvýší tak aby signalizovala neprůchodnost

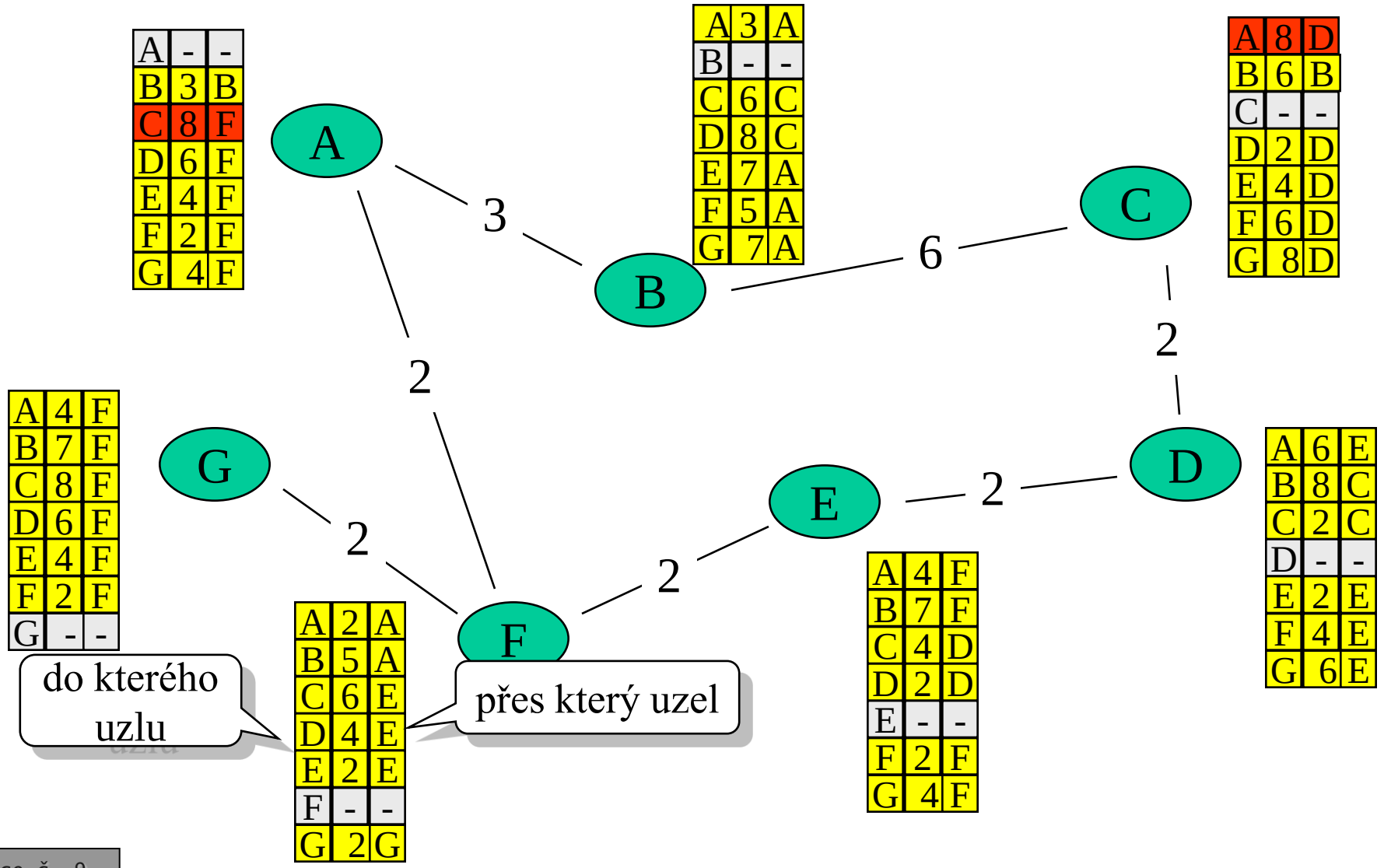
příklad - počáteční stav



příklad - stav po 1. kroku

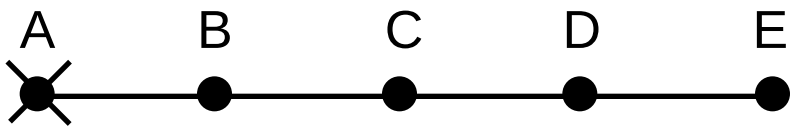


příklad - stav po 4. kroku



problém count-to-infinity

A přestalo být dostupné



1	2	3	4	původně
3	2	3	4	po 1. kroku
3	4	3	4	po 2. kroku
5	4	5	4	po 3. kroku
5	6	5	6	po 4. kroku
7	6	7	6	po 5.kroku

B si myslí, že A je ve vzdálenosti 1 v přímém směru

B zjistí, že A není dostupné přímo, ale C mu hlásí, že se "umí" dostat do A za cenu 2, proto si B poznačí že A je od něj ve vzdálenosti 3

C zjistí, že k A se dostane přes B nebo D za cenu 3, k tomu si připočte svou vzdálenost 1 od B (i D) a dostane hodnotu 4

zareagují B a D

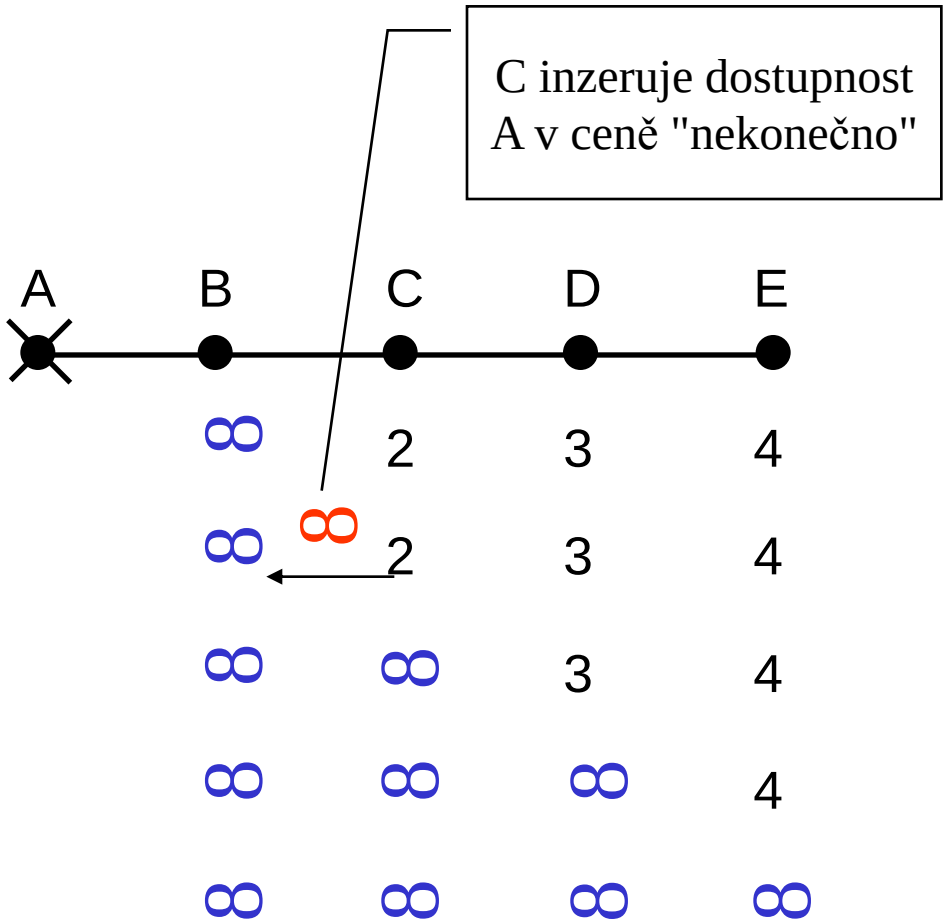
zareagují C a E

zareagují B a D

atd. až do nekonečna

řešení problému "count-to-infinity"

- metoda "split horizon"
  - směrovač nebude inzerovat "zpátky"
    - jde-li o cestu do A, kterou se Y dozví od X, pak Y nebude cestu do A zpětně inzerovat uzlu X, od kterého ji získal
- doplněk: poisoned reverse
    - uzel Y inzeruje zpět uzlu X dostupnost s hodnotou "nekonečno"
- existují takové topologie, kde i toto řešení selhává



protokol RIP (Routing Information Protocol)

- protokol typu vector-distance
  - zabudován již do BSD Unixu v roce 1982
  - metrika: počet přeskoků
    - ale jen do maxima 15!!!
    - nekonečno = 16
  - obsah směrovací tabulky (tzv. "distance vector") je rozesílán každých 30 sekund ke všem sousedním směrovačům
    - obsahuje až 25 cílových sítí
    - rozesílá se jako UDP datagram, na port č. 520
- řešení výpadků:
  - pokud není "distance vector" přijat do 180 sekund, je soused/spoj brán jako "mrtvý"
  - následně se použije metoda split horizon with poisoned reverse
- zpracování aktualizačních informací
  - řeší démon routed na úrovni OS
- použití RIP
  - je špatně škálovatelný
  - nelze jej použít pro větší sítě
  - málo stabilní

algoritmy „link-state“

- modernější, více stabilní než algoritmy vector-distance
- objemy režijních dat, které je třeba šířit po síti, jsou menší
  - a nemusí se posílat tak často
    - stačí jen při změně, nemusí se posílat pravidelně!!!!
- princip:
  - každý uzel pravidelně monitoruje průchodnost spojení ke svým sousedům
    - každou změnu distribuuje po celé síti
  - každý uzel má úplnou informaci o topologii celé sítě a o průchodnosti všech spojů
    - každý uzel si sám počítá nejkratší cesty
    - podle Dijkstrova algoritmu
      - "standardní" algoritmus pro výpočet cest v grafu
      - v praxi poněkud upravený kvůli větší robustnosti, stabilitě a konvergenci
  - každý uzel počítá "za sebe", případná chyba neovlivní ostatní uzly
    - je to náročnější na výpočetní kapacitu
- příklad:
  - protokol OSPF
    - Open Shortest Path First

dnes hojně používaný

fungování link-state algoritmů (OSPF)

- po zapnutí si každý uzel zjistí, jaké má přímé sousedy
  - pomocí paketů protokolu HELLO
- uzel průběžně zjišťuje dobu odezvy svých sousedů
  - posílá jim ECHO pakety, které se ihned vrací
- každý uzel pravidelně sestavuje paket, do kterého dá „naměřené“ hodnoty odezvy svých přímých sousedů (ohodnocení hran)
  - tento paket rozešle všem ostatním uzlům !!!!
    - prostřednictvím záplavového směrování
  - pakety stačí rozesílat jen při změně
    - po úvodním "seznámení" s topologií celé sítě
- každý směrovač postupně „naakumuluje“ zprávy o stavu všech spojů v síti,
  - díky tomu získá informace o úplné topologii sítě
  - získá také všechny informace potřebné pro výpočet cest v síti
- výpočet nejkratších cest probíhá lokálně
  - podle Dijkstrova algoritmu
    - "standardní" algoritmus pro výpočet cest v grafu
    - v praxi poněkud upravený kvůli větší robustnosti, stabilitě a konvergenci
  - každý uzel počítá "za sebe", případná chyba neovlivní ostatní uzly
    - je to náročnější na výpočetní kapacitu

velmi podstatné pro úsporu objemu aktualizačních informací !!!

srovnání

	Vector distance	Link state
Jak vnímá topologii sítě?	"pohledem svých sousedů"	vnímá celou topologii celé sítě
Způsob výpočtu cest v síti	výpočet je distribuovaný (každý něco přičte k výsledku svých sousedů)	každý si počítá všechno sám
Konvergence výpočtu	pomalá	rychlá
Chyba ve výpočtu	ovlivní ostatní směrovače	neovlivní ostatní výpočty
Aktualizace	musí být časté a pravidelné – každých 30 sekund	stačí při změně (jinak pro osvěžení každých 30 minut)
Komu se posílají aktualizační informace?	přímým sousedům	všem uzlům v síti
Škálovatelnost	špatná (max. 15 hop-ů)	lepší

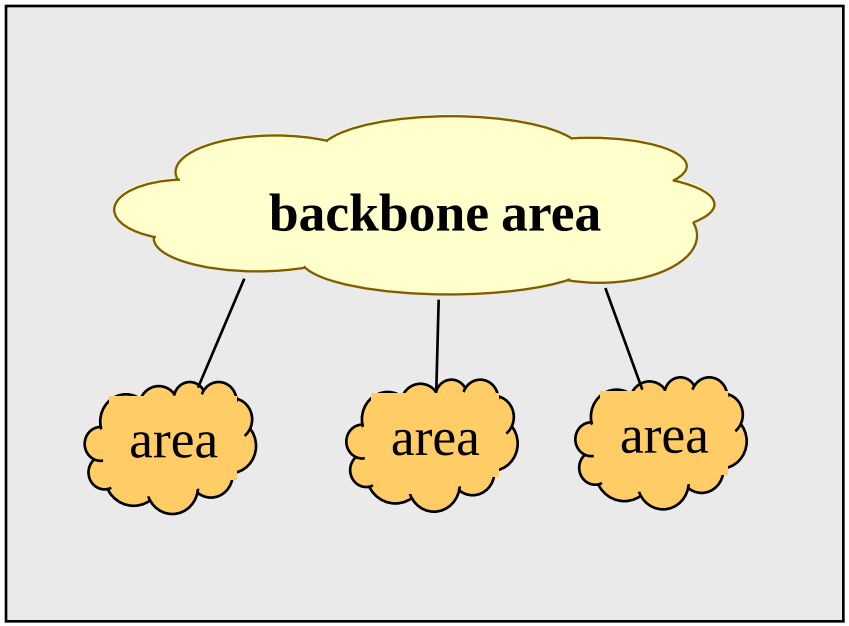
hierarchické směrování

- ani algoritmy „link-state“ nejsou vhodné pro opravdu velké sítě
- kde je problém?
  - ve velikosti sítí
    - dnes již jsou tak velké, že objemy aktualizačních informací jsou neúnosné
  - ve složitosti správy
    - směrování ve velkých soustavách vzájemně propojených sítí se stává příliš komplikované
  - v rozdílných "routovacích politikách"
    - různí provozovatelé (provideři) mohou chtít aplikovat různé strategie a koncepce ve směrování
- řešení:
  - rozdělit na menší části, s možností řešit jejich směrování autonomně
    - a nešířit podrobné směrovací informace ven
  - v rámci "menších částí" zachovat "úplné" směrování
  - do každé "části" vymezit jeden vstupní bod
    - nebo několik málo vstupních bodů
  - mezi "částmi" směrovat vždy přes jednotné vstupní body

představa hierarchického směrování

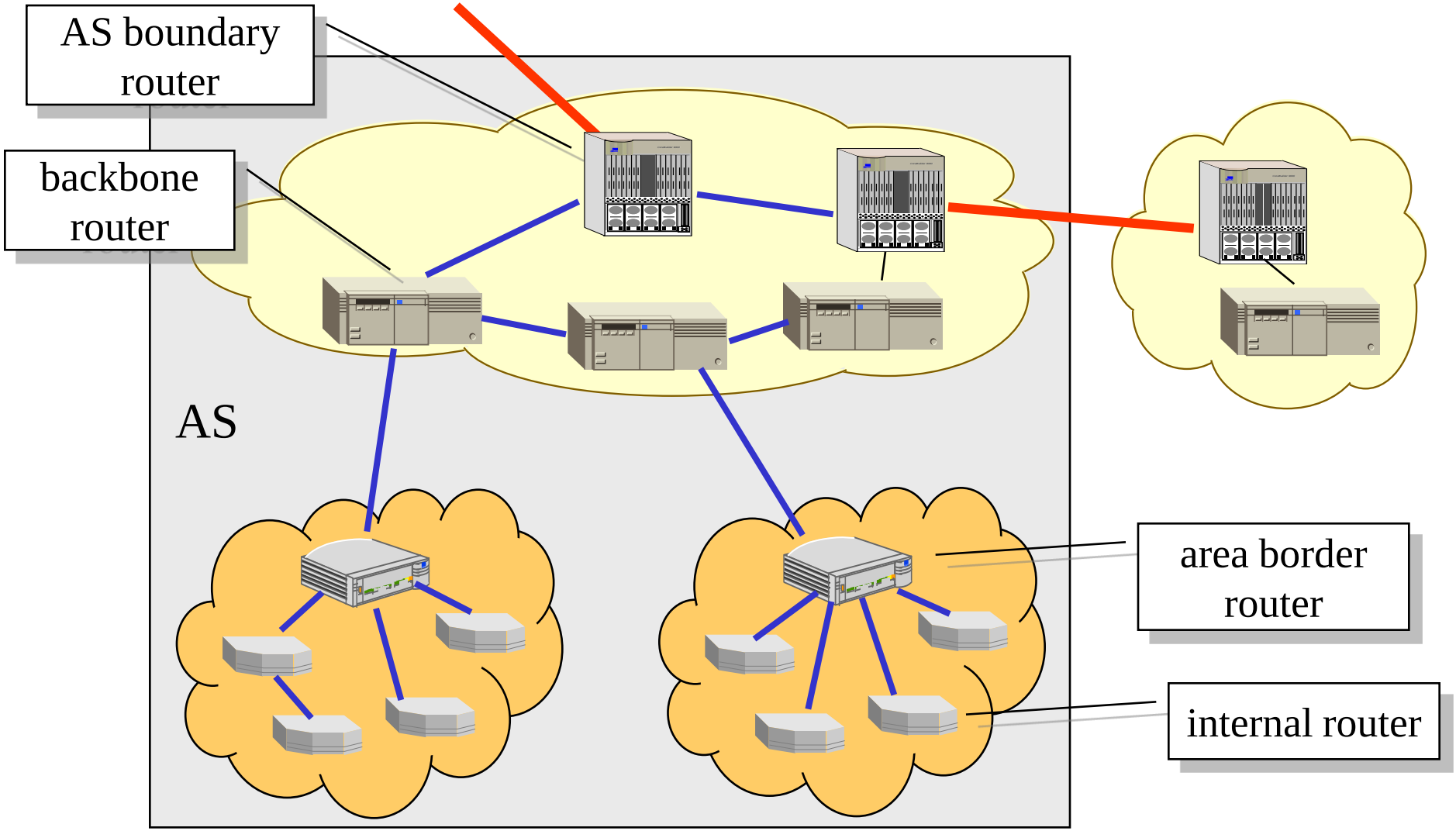
- OSPF předpokládá následující hierarchii:
  - AS
    - autonomní systém
  - backbone area
    - páteřní systém v rámci AS
  - area
    - oblast v rámci AS, připojenou k "backbone area"

AS - autonomní systém



- představa fungování:
  - *detailní směrovací informace neopouští příslušnou oblast (area) !!!*
  - komunikace s jinými oblastmi se děje pouze přes vymezené "přestupní body" a přes "nadřízené" oblasti
    - z "area" do "area" přes "backbone area", z
    - z "backbone area" do jiné "backbone area" přes autonomní systém

hierarchické směrování dle OSPF



princip fungování hierarchického směrování v OSPF

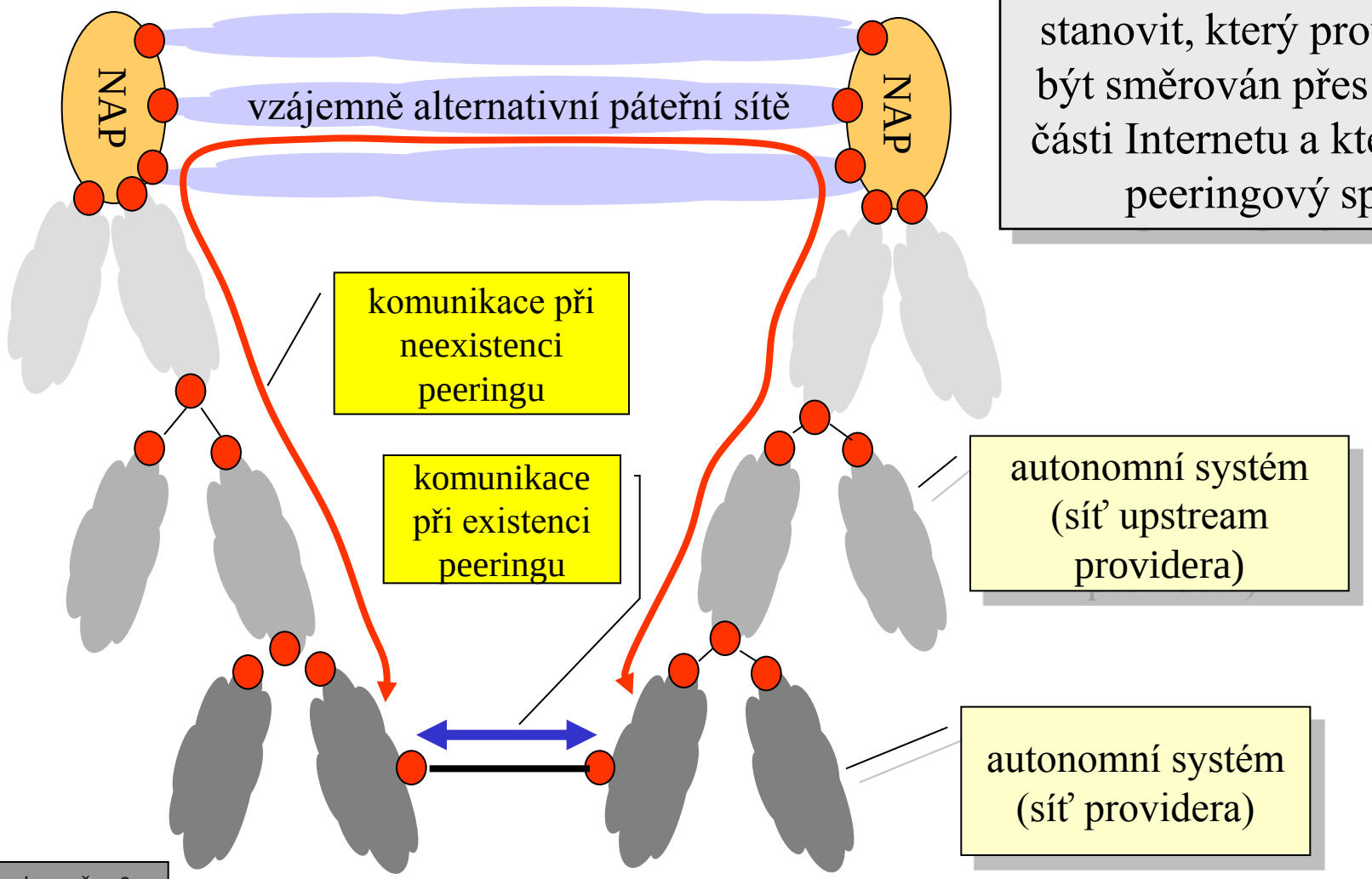
- detailní směrovací informace neopouští oblasti (area)
  - záplavové šíření aktualizačních informací se zastavuje na hranici oblasti
  - interní routery mají detailní směrovací informace o vnitřku oblasti, vše ostatní implicitně směrují na "hraniční směrovač"
- "hraniční směrovače" sumarizují informace o dostupnosti sítí v oblasti a předávají je ostatním "hraničním směrovačům"
  - informace ve smyslu: "*přes mne jsou dostupné sítě X.Y.Z až X.Y.W*"
- na úrovni páteřních oblastí se vše opakuje
  - páteřní směrovače mají detailní informace o směrování uvnitř páteře
  - mimo páteřní oblast (do jiného AS) se šíří pouze sumarizované "intervalové" informace
    - informace typu "v našem AS jsou sítě X.Y až X.Z"

tyto informace mohou být velmi "malé"

Interior vs. Exterior Gateway Protocols

- Interior Gateway Protocols:
  - provozovatelé sítí se mohou sami rozhodnout, jaké směrovací protokoly (a způsoby šíření aktualizačních informací) použijí v rámci AS i jednotlivých oblastí
  - v úvahu připadají zejména:
    - OSPF
    - RIP (dříve, dnes jen pro malé oblasti)
    - IGRP (Interior Gateway Routing Protocol, od firmy Cisco, typu vector distance)
    - EIGRP (Enhanced Interior Gateway Routing Protocol, Cisco, hybrid mezi vector distance a link state)
- Exterior Gateway Protocols
  - jsou nutné pro vzájemnou komunikaci mezi autonomními systémy (AS)
    - mezi AS boundary routers
  - umožňují definovat pravidla vzájemné komunikace mezi jednotlivými autonomními systémy
  - příklad:
    - BGP, Border Gateway Protocol

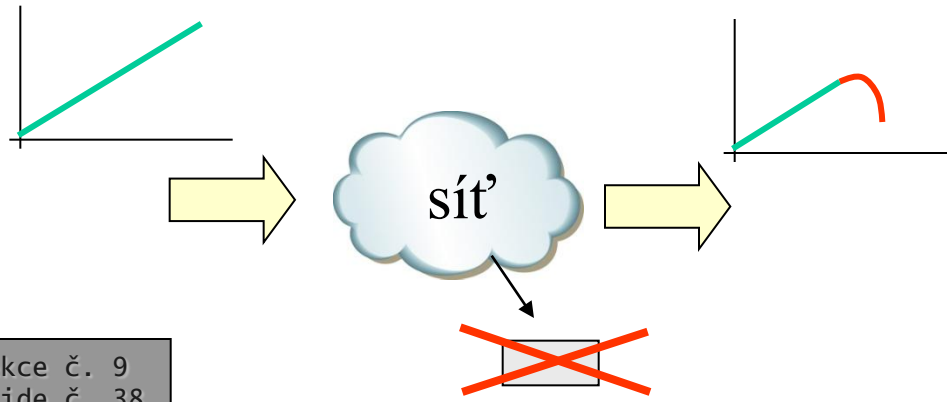
využití AS - peering



díky existenci AS je možné stanovit, který provoz má být směrován přes páteřní části Internetu a který přes peeringový spoj

předcházení zahlcení (congestion control)

- další úkol síťové vrstvy
- podobné řízení toku, ale jde o jiný problém
  - řízení toku (flow control):
    - jde o "point-to-point" záležitost, mezi jedním odesilatelem a jedním příjemcem
      - netýká se sítě mezi nimi
  - předcházení zahlcení (congestion control):
    - týká se zátěže celé sítě, datový tok od všech odesilatelů se sčítá a nesmí (neměl by) překročit hranici, kterou je síť schopna zvládnout



- zahlcení
  - stav, kdy přenosová síť musí zahazovat přenášené pakety, protože je nedokáže zpracovat (přenést)
- jak řešit?
  - "dopředné" techniky (open loop)
    - neberou v úvahu aktuální stav sítě
      - snaží se o dobrý návrh, tak aby k zahlcení vůbec nedocházelo
      - aplikují různá omezení již při odesílání, nebo v průběhu přenosu
  - "zpětnovazební" techniky (closed loop)
    - skrze zpětnou vazbou reagují na aktuální stav sítě
      - regulují odesílání podle toho, v jakém stavu je právě síť

zpětnovazební techniky

- s explicitní zpětnou vazbou
  - odesilatel dostává od sítě explicitní informaci o tom, zda došlo/nedošlo/hrozí zahlcení
    - podle toho mění své chování
- příklad:
  - síťová vrstva TCP/IP
    - datové pakety přenáší protokol IP
    - pokud došlo k zahlcení, informuje o tom odesilatele
    - informace přenáší protokol ICMP (Internet Control Message Protocol)
    - ICMP Source Quench
      - zpráva o tom, že došlo k zahlcení
      - neexistuje "protipól" (kladná zpráva, o tom že zahlcení pominulo)!!!
    - reakce odesilatele není definována
      - je ponechána na implementaci

- bez explicitní zpětné vazby
  - odesilatel sám usuzuje na to, zda došlo k zahlcení nebo nikoli
    - z jiných skutečností, například z průběhu odezvy příjemce, z míry ztrátovosti paketů atd.
- příklad:
  - protokol TCP
    - zajišťuje spolehlivý přenos
    - dostává potvrzení od příjemce
    - pokud nedostane potvrzení v časovém limitu, interpretuje to jako zahlcení !!!!
      - reaguje tak, že přechází na jednotlivé potvrzování (stop&wait)
      - teprve postupně zvyšuje zátěž, vždy na dvojnásobek, jakmile dostane kladné potvrzení (zvětšuje si okénko na dvojnásobek)

"dopředné" techniky

- řeší se snáze, pokud se používají virtuální okruhy
- příklad:
  - nově navazované virtuální okruhy se vedou mimo zahlcené části sítě
  - při navazování spojení je "uzavřen kontrakt se sítí"
    - ten, kdo navazuje spojení, říká kolik zdrojů bude od sítě potřebovat
    - síť buď požadavky akceptuje (a vyhradí si k tomu potřebné zdroje), nebo odmítne spojení navázat
    - takto to funguje např. v ATM

- triviální řešení:
  - předimenzování sítě
- traffic conditioning
  - obecně techniky ovlivňování datového toku, aby "lépe prošel" skrze přenosovou síť
  - varianta: traffic policing
    - co je nad určitý limit, je eliminováno (zahozeno)
      - již u odesilatele/po cestě
  - varianta: traffic shaping
    - snaha různě "tvarovat" datový tok
    - například chvíli pozdržet pakety, aby byly odesílány s rovnoměrnými odstupy

používá se například u českého ADSL