

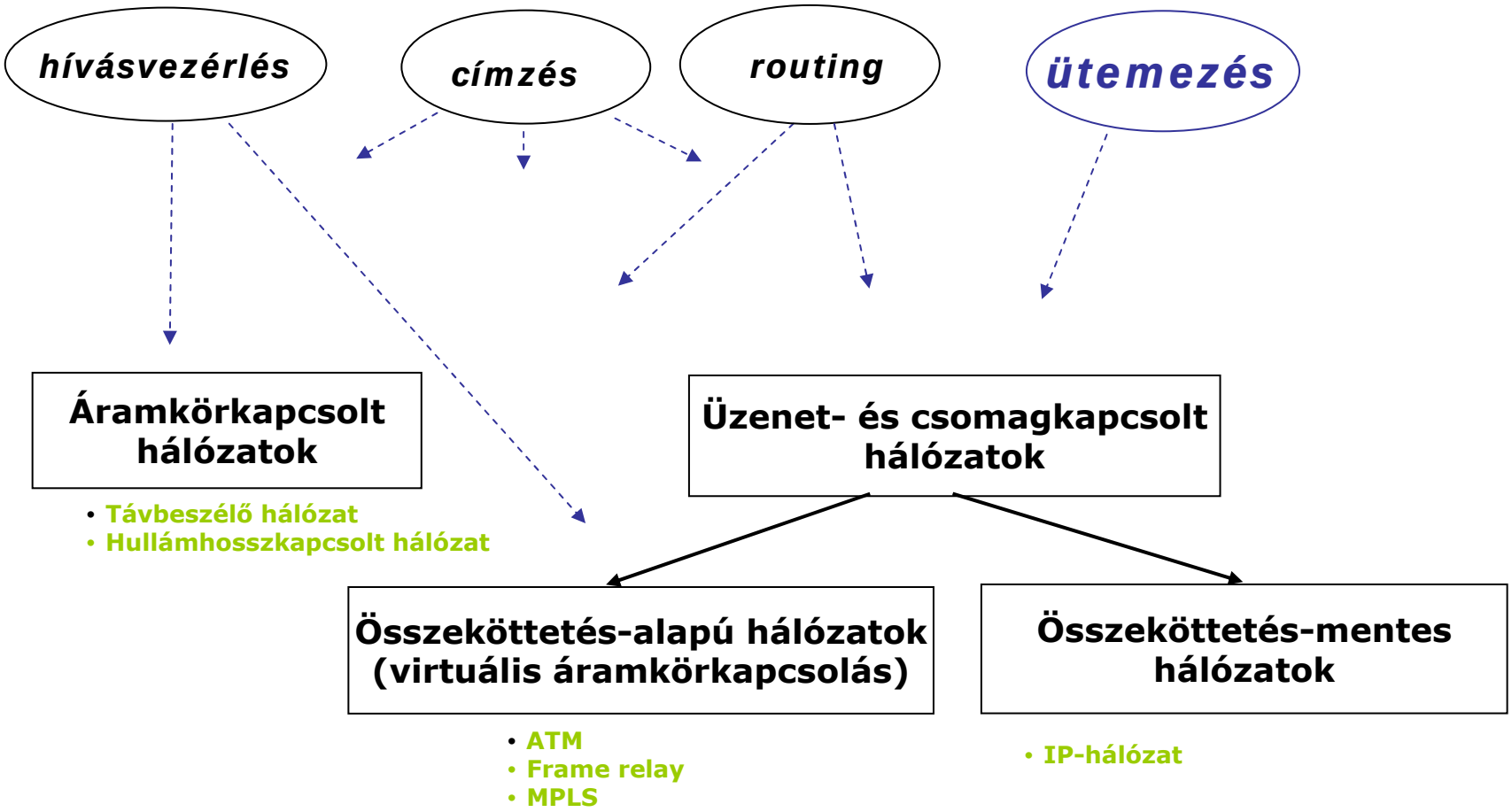
FELADATÜTEMEZÉS, CSOMAGKEZELÉS

Scheduling

Dr. Simon Vilmos
adjunktus
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu

2013. március 27.

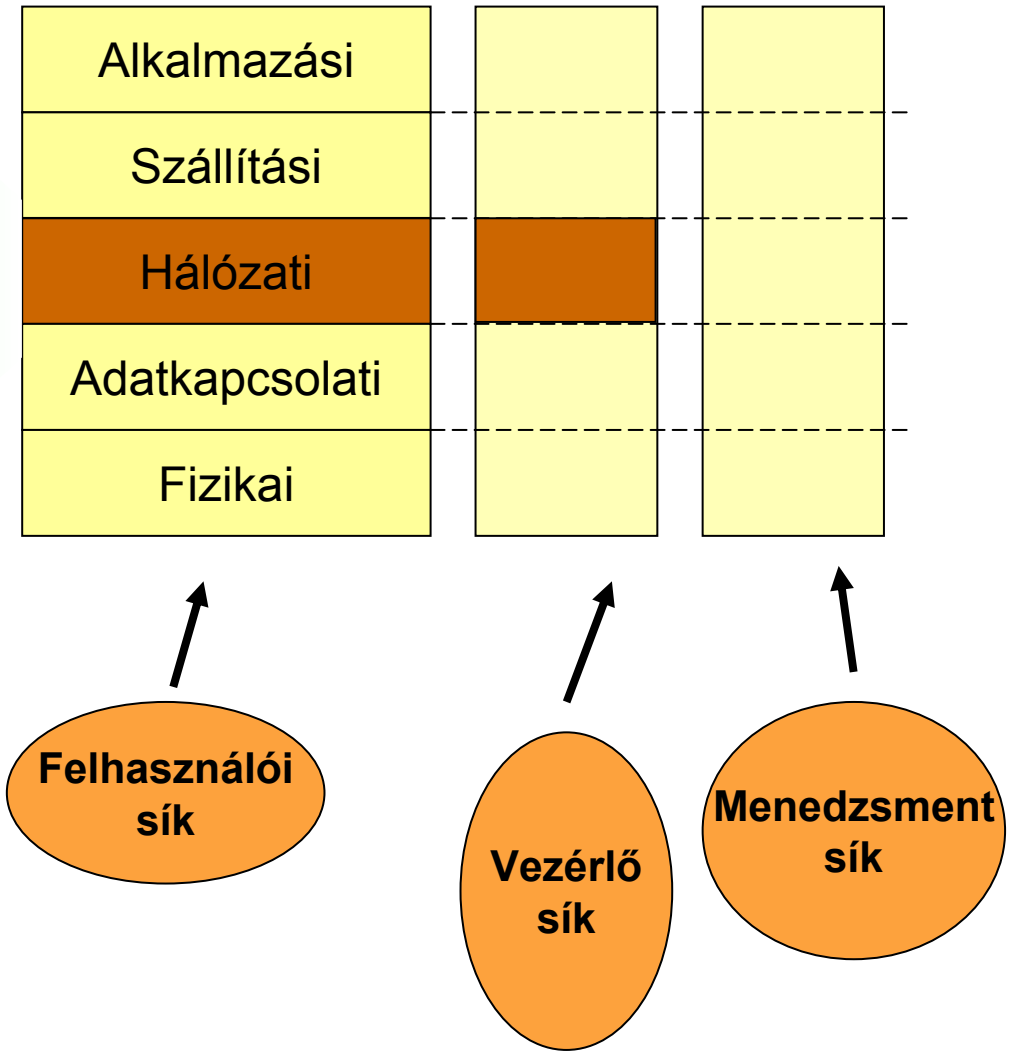
További nagyon fontos funkció a kapcsolt hálózatok működtetéséhez



Elv:

Ütemezés (packet scheduling)

Módszerek:
GPS
Weighted round robin
Deficit round robin
WFQ



Meg kell jegyezni, hogy cross-layer megoldásai is vannak!

Mi az, hogy „scheduling”, ütemezés?

- Több területen is megjelenő problémára megoldás
- **Véges erőforrás:**
  - **időnként nagyobb mértékben kellene igénybe venni**, mint amekkorára az képes
  - máskor viszont nincs eléggé leterhelve
  - **Ha igaz, hogy hosszú idejű átlagban képes az igények kiszolgálására: ütemezés!**

    *(ha nem akkor **erőforrás bővítés** kell)*
- Időnként tehát **várakozni kell** az erőforrásra
  - Milyen **elvek, stratégiák alapján** képezzünk sort, sorokat a várakozókból, ahhoz, hogy
  - **adott kiszolgálási feltételek, elvárások** a lehető legjobban teljesüljenek

Scheduling a hálózatokban

- Csomagkezelés a hálózat csomópontjaiban
  - **Csomag jön → cél-cím → csomag megy**

- Véges erőforrások:
  - **A linkek átviteli képessége**
  - **A csomópontok tárolási képessége**
    - **tárolás, sorbaállítás, átrendezés**
  - **A csomópontok feldolgozási képessége**

- Jól van méretezve az a hálózat, amelynél ez előfordul?
  - nagyszámú igény adja össze a teljes igénybevételt
  - általában kiegyenlítik egymást
  - eltekintve kilógó esetekről, amelyekre jó becslést lehet adni, ez a
  - **statisztikus multiplexelés**
- és így jobban járunk, mintha a csúcsra méreteznénk ez a
  - **statisztikus nyereség**

Miről lesz szó ebben a részben?

- *Bevezetés*
- *Követelmények*
- *Lehetőségek*
- *A „best effort” kiszolgálás ütemezői*
- *Ütemezés garantált kiszolgálás esetén*
- *Csomageldobás*

Bevezetés

- **Verseny** az osztott használatú erőforrásokért
  - Miért verseny, miért nem FCFS (**f**irst-**c**ome-**f**irst-**s**erved)?

- Feladatütemezési módszer kell:
  - Az **igazságos megosztáshoz**
    - lehet egyenlő, de nem feltétlenül az egyenlősdi az igazságos
  - A **kiszolgálási minőség** garantálásához
    - különböző típusú forgalomnak különböző elvárásai vannak

- A módszer két független összetevője:

  - **Kiszolgálási sorrend meghatározása - késleltetés**
  - **Kiszolgálásra várakozók túlcsordulásánál igényeldobási „stratégia” - vesztési arány**

Miről van szó tehát, amikor ütemezésről beszélünk hálózatokban?

- A hálózatokban osztott:
  - az átviteli képesség
  - a csomópontok tárolói
- A feladatütemezés helye:
  - a csomópontok tárolóiban
- A forgalom statisztikus ingadozását kezeljük az ütemezéssel

Miért van szükség (nem triviális) ütemezésre?

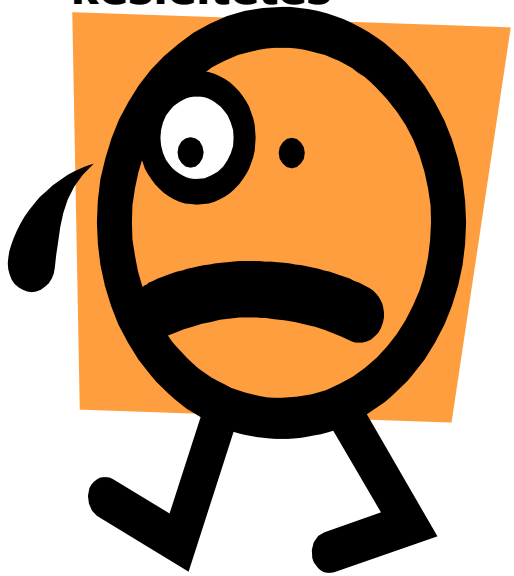
- **Legalább** kétféle alkalmazást kell kiszolgálni
  - ***Rugalmas (elasztikus) vagy késleltetéstűrő alkalmazások***
    - *elviselik a véges és változó ábocsátóképességet (pl. file-átvitel)*
    - igazságos megosztás
    - *„best effort" igény* (~ legjobb szándékú)
  - ***Merev, vagy intoleráns alkalmazások***
    - *garantált szolgáltatást igényelnek*
      - *pl. a beszéd állandó 64 kbit/s-os csatornát*
    - korlát a késleltetésre, garantált átviteli sebesség, korlát a vesztési arányra
    - *garantált szolgáltatású igény*

Megőrzési törvény (Conservation Law)

- Triviális ütemezés: FCFS
- de mindenfajtára igaz, hogy:
  az átlagos késleltetések forgalom részaránnyal súlyozott összege = konstans
- *Valakinek előnyt csak mások rovására lehet biztosítani*
- Bármely ***munkamegőrző* (work-conserving)** ütemező csak ugyanazt az összeget osztogathatja
- „Szorgalmas” vs. „Lusta” ütemező

$$\sum_{i=1}^{N} \rho_i \cdot q_i = \text{konstar}$$

forgalom-részarány **átlagos késleltetés**



Alapvető követelmények

- Egyszerű megvalósíthatóság
- „Best effort” kiszolgálásnál:
  - **Igazságosság (fairness)** biztosítása
  - **Védelem (protection)** biztosítása
- Garantált kiszolgálásnál:
  - **Teljesítménykorlátok (performance bounds)** biztosítása
  - Egyszerű és hatékony **beengedés-szabályozás (admission control)**

Egyszerű megvalósíthatóság

- Nagysebességű hálózat → gyakori csomagtovábbítás
- Csomagonként csak néhány műveletre van lehetőség
- Ha *N* a kiszolgálandó igények száma:
  - a csomagonkénti feldolgozási időigény **ne exponenciálisan**, hanem **lineárisan** nőjön *N* függvényében
- Elsődleges korlát az állapotok nyilvántartása (pl. pointer a sorban):
  - az ehhez szükséges memória és
  - elérési idő

Igazságosság és védelem

- Erőforrás elosztás *igazságos* legyen:
  - A részesedés **a költségviselés arányában** történjen
- Amennyiben valaki megkísérel **jogosulatlan előnyhöz** jutni, legyen megakadályozva ebben → *védelem* a többieknek

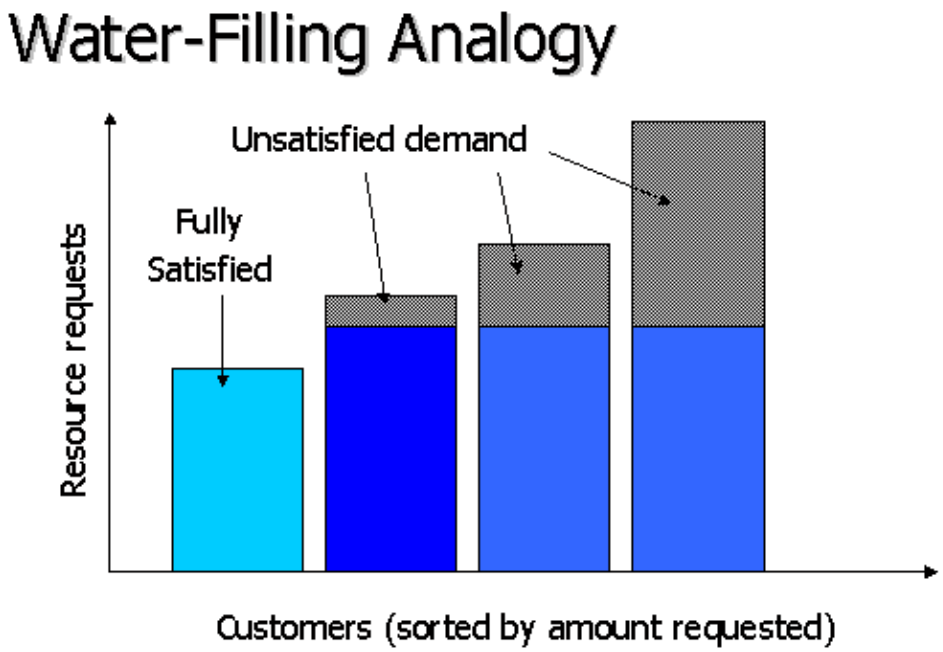
Igazságosság és védelem

- Gyakori eset: egyenlő jogosultságok, de vannak, amelyek igénye kisebb a többiekénél
  - logikus, hogy a „kis” felhasználóknak adjuk oda, amennyit akarnak
  - a fennmaradót igazságosan a nagyok között
- Ez a max-min igazságos megosztás elve:
  - Erőforrás-kiosztás az igények növekedése szerint
  - Senki sem kap többet a kértnél
  - A kielégítetlen igények egyenlően osztoznak a maradékon
- Ezt formalizálva, kapjuk a **max-min algoritmust**

Példa max-min algoritmusra

- 4 forrás: 2; 2,6; 4; 5 igényekkel
- 10 kapacitású erőforrással.
  - **1. lépés: mindenki 2,5.**
  - **a 0,5-öt egyenlően elosztjuk a 3 között, ez ~2,66-ot ad**
  - **a ~0,06-ot a 2 között,**
  - **végül az 1-es 2-t, a 2-es 2,6-ot, a 3-as és 4-es ~2,7-et kap**

Max-min szemléltetése



Eltérő jogosultságok: súlyozott max-min megosztás

- Például 15 egységnyi erőforrásra 5 pályázó

A	B	C	D	E
1/15	2/15	3/15	4/15	5/15
2	4	7	5	2
1	2	3	4	5
				+3

- Jogosultságaik aránya
- Igényeik:
- Normalizált súlyok:
- Az E kivételével mindenki elégedetlen:
- Szétosztjuk súlyaik arányában, D is OK:

1,3+2,6+3,9+5,2+2=15

+0,2

Ez még szétosztható

Igazságosság és átbocsátóképesség

- Eljárások hol nem teljesül az igazságosság (fairness):
  - Néha magas átbocsátóképesség
  - De instabil szolgáltatásminőség
    - Felhasználók viselkedésének függvényében
  - Ha gyakran instabil: elégedetlen felhasználók

Max-min előnyei

- Összehasonlítva az egyenlő felosztással:
  - Jobb erőforráskihasználtság
    - mivel egyenlő felosztás esetén nem használjuk az egyes felhasználók maradékát
- Egyéb előnyök:
  - „Rosszul” viselkedő (nagy méretű csomagok vagy börsztös átvitel) felhasználók magukat büntetik, nem mást
- Ezt az elvet használja (lásd később):
  - Round robin
  - Far Queuing

Maximális átbocsátóképességű ütemezés

- **Maximum throughput scheduling**
- Prioritás azon folyamok számára, amelyek a „legolcsóbbak” a hálózat szempontjából
  - „Legolcsóbb”: legkisebb hálózati erőforrás/átvitt információ
  - Pl. rádiós hálózatoknál: prioritás az alacsony csillapítású felhasználóknak (magas SNR)
    - „Drága” felhasználó: távol van a bázisállomástól
- Költségfüggvényt számol, pl. csillapítást felhasználókhoz
- Legjobb erőforrás kihasználás, de !

Gondok a maximális átbocsátóképességű ütemezéssel

- Többi felhasználó folyamának meg kell várni míg a „legolcsóbbak” kiszolgálásra kerülnek
  - **Éheztetés (starvation)**

- Hiába ad jobb erőforrás kihasználást mint a max-min, nem teljesül a fairness elv
  - Elégedetlen felhasználók
  - Szolgáltató profitja ettől is függ

Proportional fair

- Kompromisszum a max-min és maximális átbocs.ü. között
- Előnye van az „olcsóbb” folyamoknak, de „drágábbak” is kapnak egy adott kiszolgálási szintet
- Minden felhasználóhoz egy prioritási szint
  - Fordítottan arányos a várható erőforrás használatukkal
  - Pl. Weighted Far Queuing (lásd később)

„Merev” alkalmazások: teljesítménykorlátok biztosítása

- Teljesítmény, teljesítőképesség - *performance*
- Akár felhasználónkénti kiszolgálási **teljesítménykorlát garantálása**
  - a megőrzési törvényen belül
- Szerződés: felhasználó ↔ szolgáltató
- Egyrészt kiszolgálási garancia, másrészt használati kötelezettségvállalás
- Garancia nem csak egyetlen ütemezőre, hanem elvileg az egész hálózatra
  - **végpontok közötti garancia, igen nehéz feladat**
  - **mivel ütemezés csomópontokon**

Teljesítménykorlátok fajtái

- Determinisztikus:
  - Az összeköttetés valamennyi csomagjára teljesülnie kell

  Egyszerű ellenőrzés, de rossz kihasználtság

- Statisztikus:
  - A csomagok adott hányadára teljesül
  - *N* egymás utáni csomagból egyre nem teljesül

  Bonyolult ellenőrzés, de jó kihasználtság

Gyakran használt teljesítménykorlátok (teljesítőképesség-jellemzők, QoS-paraméterek)

- Sávszélesség
  - garantált minimális sávszélesség az összeköttetésre
- Késleltetés
  - Legrosszabb eset
    - minden más összeköttetés a lehető legrosszabbul „viselkedik”
  - Átlagos érték
    - az adott összeköttetés csomagjainak késleltetését átlagoljuk
  - Csomagok adott hányadára vonatkozó jellemző
    - pl. a csomagok 99%-a kisebb késleltetésű lesz, mint a 99-percentilis késleltetésérték
- Késleltetés-ingadozás
  - kiegyenlítési lehetőség a vevőben, de nem lehet akármekkora
- Csomagvesztés

Egyszerű és hatékony beengedésszabályozás

- A teljesítménykorlátok (QoS paraméterek) csak **beengedésszabályozás (admission control)** alkalmazásával biztosíthatók
- Gyorsan kell eldönteni, hogy újabb igény még kiszolgálható-e
- Nem eredményezheti azt, hogy a hálózat nem lesz kihasználva a lehetséges mértékben
- Például FCFS esetén
  - Legnagyobb késleltetésre garancia:
    - Létszámkorlát és limitált börszt-méret

    Kis kihasználtság!

Miről volt szó eddig és mi következik?

- *Bevezetés*
  - *fogalmak, a megőrzési törvény*
- *Követelmények*
  - *megvalósíthatóság*
  - *igazságosság*
  - *teljesítménykorlátok, beengedésszabályozás*
- *Lehetőségek*
  - *prioritások...*
- *A „best effort” kiszolgálás ütemezői*
  - *GPS elv*
  - *Round-robin, súlyozott round-robin*
  - *Weighted Fair Queueing*
- *Ütemezés garantált kiszolgálás esetén*
- *Csomageldobási stratégiák*

Alapvető választási lehetőségek az ütemezés tervezésénél

- Négy alapvető szabadsági fokunk van a tervezésnél
  - Prioritási szintek száma (A)
  - Az egyes szinteken
    - munkamegőrző (work-conserving) vagy nem munkamegőrző (non-work-conserving) módot használjunk-e? (B)
  - Igények csoportosítási (aggregálási) mértéke az egyes szinteken belül (C)
  - Kiszolgálási sorrend az egyes szinteken belül (D)

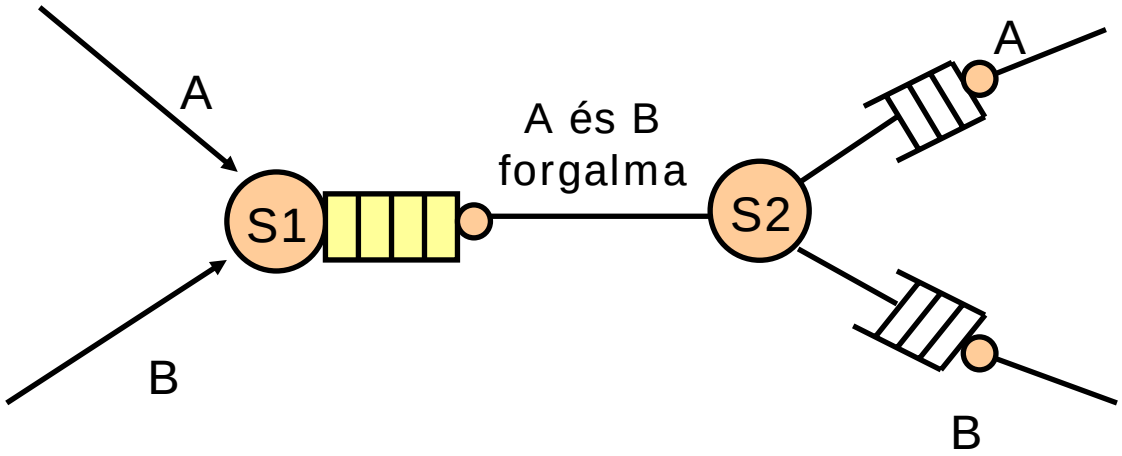
(A) Prioritás

- *n* szint esetén a *k*-adik szintű igényt akkor elégíti ki csak az ütemező, ha nincs kiszolgálásra váró igény a *k+1, k+2, …, n* szinteken
- Magasabb prioritási szint = kisebb késleltetés
- Egyszerű a megvalósítása
- Jól számolható a teljesítőképessége
- Probléma: minden áron előnyben részesítés, a magasabb szintű igény **„kiéhezteti”** az alacsonyabb szintűeket
- Megoldás: megfelelő beengedésszabályozás
- **Gyakorlatban: max. 3 prioritási szint**

(B) Work-conserving és non-work-conserving kiszolgálás („szorgalmas” vagy „lusta” kiszolgáló)

- **Work-conserving**: csak akkor tétlen a kiszolgáló, ha nincs várakozó igény (csomag)
- **Non-work-conserving**: vannak tétlen időszakai akkor is, ha van várakozó igény
- Miért jöhet egyáltalán szóba a „lusta”?
- A kivárás előnyös lehet a forgalmi jellemzőkre
- Csak az „esedékessé” váló csomagokat továbbítjuk
  - Csökken a tároló iránti igény
  - Csökken a késleltetés ingadozás
  - Egyszerűsödik a vállalható teljesítménykorlát meghatározása
  - De nagyobb teljes késleltetés

Példa: amikor a nem munkamegőrző kiszolgálás hasznos lehet



- „A” beérkezési folyamata az S2 kimeneti pufferébe attól is függ, milyen „B” viselkedése az S1-nél
- „B” forgalma feltartóztathatja „A” csomagjait, így amikor azok végre elhagyják S1-et, csomósodva érkeznek meg az S2 kimenetére

Nem-munkamegőrző kiszolgáló

- Az esedékessé válás időpontjainak meghatározása
  - pl. a „rate descriptor” alapján
- Szükség van-e ilyen kiszolgálóra?
  - Viszonylag új technika, egy évtizeddel ezelőtt még kutatási kérdés volt
  - Értékelés:
    - csökkenti a *késleltetés ingadozást* az átlagos késleltetés megnövelése árán és a *kapcsolók tárigényét*
    - **korrekt forgalomjellemzést** kíván meg a forrásoktól, és azt, hogy **ahhoz tartsák is magukat**

Nem-munkamegőrző kiszolgáló alkalmazása

- Csomagformázás (packet shaping):
  - Korlátozza a folyam „kijátszásának” rátáját a kimenő interfészre
  - Pl. multimédia forgalom esetén néha nem éri meg siettetni a továbítást
    - Hang nem lesz előbb kijátszva, mint ahogy felvették
- Gyakorlatban nem alkalmazzák
- Lehetséges ötvözni a munkamegőrzővel
  - Munkamegőrző használja a kapacitását ameddig a nem-munkamegőrző szünetel

Hol tartottunk: tervezési lehetőségek

- Négy alapvető szabadsági fokunk van a tervezésnél
  - Prioritási szintek száma (A)
  - Az egyes szinteken
    - munkamegőrző (work-conserving) vagy nem munkamegőrző (non-work-conserving) módot használjunk-e? (B)
  - Igények csoportosítási (aggregálási) mértéke az egyes szinteken belül (C)
  - Kiszolgálási sorrend az egyes szinteken belül (D)

(C) Igények csoportosítása

- Összeköttetések összevonása (aggregation) és közös kezelése
- Két véglet:
  - **valamennyi igény együttes jellemzése** – ugyanazt a szolgáltatásminőséget kapják
  - **minden egyes összeköttetésre saját QoS** biztosítása
- Csomagkapcsolt hálózaton technikailag nem lehet erőforrással győzni az egyedi igények kezelését
  - szemben a telefonhálózattal, ahol azonosak az igények
  - mindenekelőtt a kapcsolók ütemezői által kezelt állapotváltozók mennyisége a kritikus
    - ha összeköttetésenkénti, akkor megengedhetetlenül nagy lehet

Igények csoportosítása: osztályba sorolás

- Közbenső eset: osztályokba sorolás
  - az adott osztályba sorolt összeköttetések ugyanazt a szolgáltatásminőséget kapják
- Nincsenek védve egymástól
  - mivel ütemező nem tud különbséget tenni az osztály összeköttetései között, ha egy megsérti a „szerződést”, mindenki rosszabb kiszolgálást kap → **kiszolgáltatottság**
- Torlódás kezelése
  - mivel ..., ha egy csoporttag okozza is a torlódást, mindegyik jelzést kap, hogy fogja vissza magát
- Jó lenne megoldani, hogy csak az értelmezze, amelyik okozta

(D) Kiszolgálási sorrend (prioritási szinten és csoporton belül)

- Érkezési sorrendben történő kiszolgálás (FCFS) ↔ érkezéstől eltérő sorrendű kiszolgálás
- A FCFS hátrányai:
  - Nem engedi meg a kivételt, pl. késleltetés érzékeny összeköttetések csomagja számára
  - Nem biztosít védelmet, nem *max-min* elvű
  - Erőszakosságra ösztönöz

- A nem-FCFS ütemező előnyös, de bonyolult
  - Meg kell határozni, és jelezni a sorrendet (tagging)

A „best effort” kiszolgálás ütemezői

- Általánosított processzormegosztás (GPS) - elmélet
- Súlyozott „round-robin” – innentől gyak. közelítések
- „Deficit round-robin”
- A WFQ – weighted fair queueing *vagy*
  PGPS – Packet-by-packet GPS
  ~ súlyozott igazságos sorképzés ill.
  csomagonkénti GPS

A GPS (Generalized Processor Sharing) elve

- A **max-min** igazságosság megvalósítása
- Egy **alapelv** kimondása csupán: úgy lehetne igazságosan megosztani az erőforrást, ha sorban mindenkinek egy 0-hoz tartóan kicsiny elemi „szeletét” végeznénk el a feladatából
- **Elvileg megoldjuk** a GPS-szel az igazságos kiszolgálás problémáját, de mit lehet tenni **a gyakorlatban**?
- **A GPS közelítéseit** alkalmazhatjuk, és a GPS-hez viszonyíthatjuk azok tulajdonságait

A GPS

- Mintha mindegyik csomag **külön logikai sorban** lenne
- Mindenki **egymás után kap egy parányi kiszolgálást**, majd **„körben”** folytatódik
- Akinek nincs igénye az kimarad
- Különböző jogosultságok esetén a nullához tartóan kis kiszolgálás a súlyok arányában különböző lesz
- Kimutatható, hogy a GPS max-min értelemben **fair kiszolgálást** nyújt
- A GPS egy **absztrakció**
- A kérdés: mennyire közelíti egy valóságos ütemező a GPS-t?

Súlyozott körbekérdezés (Weighted round-robin)

- A round-robin a GPS legegyszerűbb közelítése
- Jó, ha azonos csomaghosszak és azonos súlyú összeköttetések
- Kiszolgálás körben **csomagonként,**
  - különböző súlyokkal,
  - különböző csomaghosszakkal

Hátrányai

- Átlagos csomaghossz jó közelítésű ismerete kell
  - IP hálózatokban ez nem könnyű, változó csomagméretek!
- *Rövid időszakra nagyon igazságtalan lehet*

Példa súlyozott körbekérdezésre

- Pl. három összeköttetés:
  A, B, C
- átlagos csomaghosszak:
  50, 500, 1500 byte
- súlyok: 0,5:0,75:1
- Hány csomagot/byte-ot szolgáljunk ki az egyes összeköttetésekből egy körben?
- súly/csomaghossz:
  1/100; 3/2000; 1/1500
- normalizált súly/csomaghossz:
  60, 9, 4
  (3000, 4500, 6000 byte)

„Deficit round-robin”

- Előre ismeretlen átlagos csomaghosszra
- Definiálunk
  - egy ***kiszolgálási adagot***, *kvantumot* (byte-okban mérve)
  - egy ***számlálót***, amely a felhasználó „hitelét” számolja
- A sor elején álló csomagot akkor szolgáljuk ki, ha az nem hosszabb, mint az adag
  - ha hosszabb, az adagot hozzáadjuk a számlálóhoz
  - és a következő körben ennek vizsgálata alapján döntünk

Példa deficit round robinra

- *Példa:*

		A	B	C
Kiszolgálási adag: 100				
Csomaghossz:		*150*	*80*	*120*
Hitelszámláló:	*1. kör*	*100*	*20*	*100*
	2. kör	*50*	*0*	*80*

- *1. kör: B (marad 20 hitele), A és C kimarad*
- *2. kör: A és C, mert*
  megmaradó hitel: hitel + adag – csomaghossz

- *akinek nincs igénye, az elveszti a hitelt*

A WFQ (Weighted Fair Queueing)

- Alapötlet: kiszámítjuk (szimuláljuk) a csomagok távozási időpontjait, mintha GPS szerint szolgáltuk volna ki azokat, és ezt a sorrendet alkalmazzuk a kiszolgálásra
- Nem a távozási időpont, hanem a sorrend érdekes
  - **ezért a távozást jellemző értéket így *befejezési számnak* (*finish number*) nevezik**
- (Legyen bitenkénti kiszolgálás)
- **Ciklusszám** (***round number***) – pozitív nem egész szám
  - az aktív felhasználók számának reciprokával arányos
- **Ciklushossz:** arányos az aktív felhasználók számával
- Ha ismerjük a ciklusszámot, a **befejezési szám** kiszámolható:
- $F(i,k,t) = \max[F(i,k-1,t),R(t)]+P(i,k,t)$
  - $F(i,k,t)$ – az ***i***-edik felhasználó befejezési száma ***t*** időpontban
  - $P(i,k,t)$ – az ***i***-edik felh. ***t***-ben beérkező ***k***-adik csomagjának hossza
  - $R(t)$ – ciklusszám a ***t***-ben

Hogy jön ki a WFQ összefüggés?

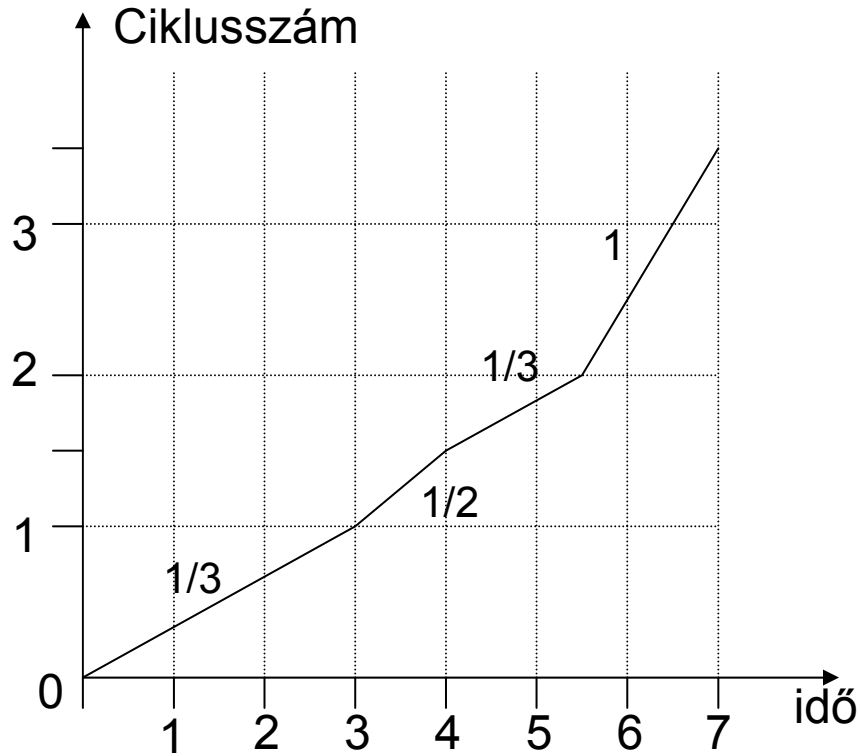
- *Inaktív* felhasználónál a befejezési szám:
  - az aktuális ciklusszám plusz a csomagméret
  - pl. ha egy 10 bites csomag érkezik, amikor a ciklusszám 3, a kiszolgálása akkor befejeződik be, amikor a ciklusszám 13 lesz

- *Aktív* felhasználó csomagja beérkezésekor annak befejezési száma:
  - a sorában lévő csomagok közül a legnagyobb befejezési számú plusz a csomagméret
  - pl. ha a 10 bites csomag beérkezésekor a sorban van egy 20-as befejezési számú, akkor 30 lesz
- Kombinálva ezt a két állítást:

$F(i,k,t) = \max[F(i,k-1,t),R(t)]+P(i,k,t)$

Példa a WFQ működésére
$F(i,k,t) = \max[F(i,k-1,t),R(t)]+P(i,k,t)$

- A, B, C, 1; 2; 2 a csomaghosszak t=0-ban és A még generál egy 2 hosszút t=4-ben
- Kiszolgálás: 1 csomageység/időegység
- $R(0)=0$
- $F(A,1,0) = \max[F(A,0,0),R(0)]+ P(A,1,0)=\max[0,0]+1=1$
- $F(B,1,0) = F(C,1,0) = 2$
- *A* első csomagjával kezdjük
- majd B vagy C, ami t=3-ban fejeződik be
- a harmadikat t=3-ban kezdjük és 5-ben fejeződik be
- t=4-ben *A* második csomagja
- *A* második csomag finish no-hoz kell tudnunk a round no-t, t=4-ben
- A görbe menetéből...



A WFQ értékelése

- Súlyozott esetben a befejezési szám:

  $F(i,k,t) = \max[F(i,k-1,t),R(t)]+P(i,k,t)/w(\mathrm{i})$

- Az aktuális ciklusszám meghatározása jelent gondot:

  egy kieső (kiszolgált) felhasználó megváltoztatja a ciklus-szám változási sebességét → felgyorsul a kiszolgálás → „láncreakció” következik/-het be

- Egyre általánosabb a használata a korszerű routerekben

Csomageldobás

- Nem csak a kiszolgálási sorrendről, hanem a tárolhatatlan csomagok kezeléséről is dönteni kell
  - csomageldobás
- Milyen alapon működjön a csomageldobási stratégia?
  - Csoportosítás (aggregálás): összeköttetésenként, vagy csoportonként
- Eldobási prioritások
  - Előbb dobja el az alacsony prioritású csomagokat
- Korai eldobási stratégiák
  - Early Random Drop
  - Random Early Detection

Early Random Drop és Random Early Detection

- Korai véletlen eldobás
  - Sorhossz nagyobb, mint *eldobási küszöb* →
    - azonos és állandó valószínűségű eldobás

- Véletlen korai detektálás
  - küszöb az átlagsorhosszra
    - börsztökre nem érzékeny
  - az eldobás valószínűsége arányos a sorhosszal
  - nem-kooperatív felhasználók csomagjainak jelölése

Miről volt szó a scheduling részben?

- *Bevezetés*
  - *fogalmak, a megőrzési törvény*
- *Követelmények*
  - *megvalósíthatóság*
  - *igazságosság*
  - *teljesítménykorlátok, beengedésszabályozás*
- *Lehetőségek*
  - *prioritások...*
- *A „best effort” kiszolgálás ütemezői*
  - *GPS elv*
  - *Round-robin, súlyozott round-robin*
  - *Weighted Fair Queueing*
- *Ütemezés garantált kiszolgálás esetén*
- *Csomageldobási stratégiák*

Kérdések?

KÖSZÖNÖM A FIGYELMET!

Dr. Simon Vilmos
adjunktus
BME Hálózati Rendszerek és Szolgáltatások Tanszék
svilmos@hit.bme.hu