

Számítógépes Hálózatok ősz 2006

7. Adatkapcsolati réteg – MAC, Statikus multiplexálás, (slotted) Aloha, CSMA

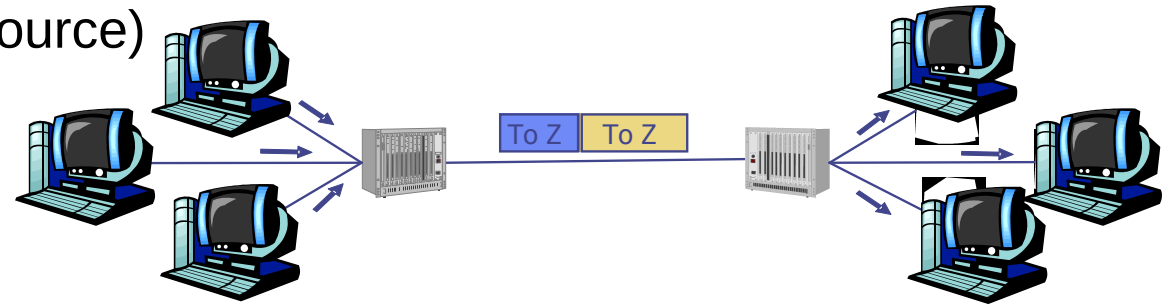
Mediumhozzáférés (Medium Access Control -- MAC) alréteg az adatkapcsolati rétegben

- **Statikus multiplexálás**
- Dinamikus csatorna foglalás
 - Kollízió alapú protokollok
 - Verseny-mentes protokollok (contention-free)
 - Protokollok korlátozott versennyel (limited contention)
- Az Ethernet példája

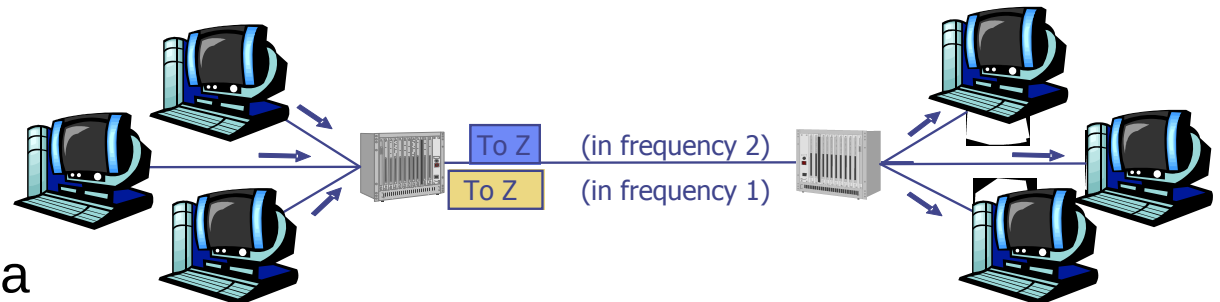
Statikus multiplexálás

- Adott egy link (erőforrás / ressource)

- A kommunikációs kapcsolatokhoz fix időegységeket (TDM) / frekvenciasávot (FDM) / csatornákat rendelünk

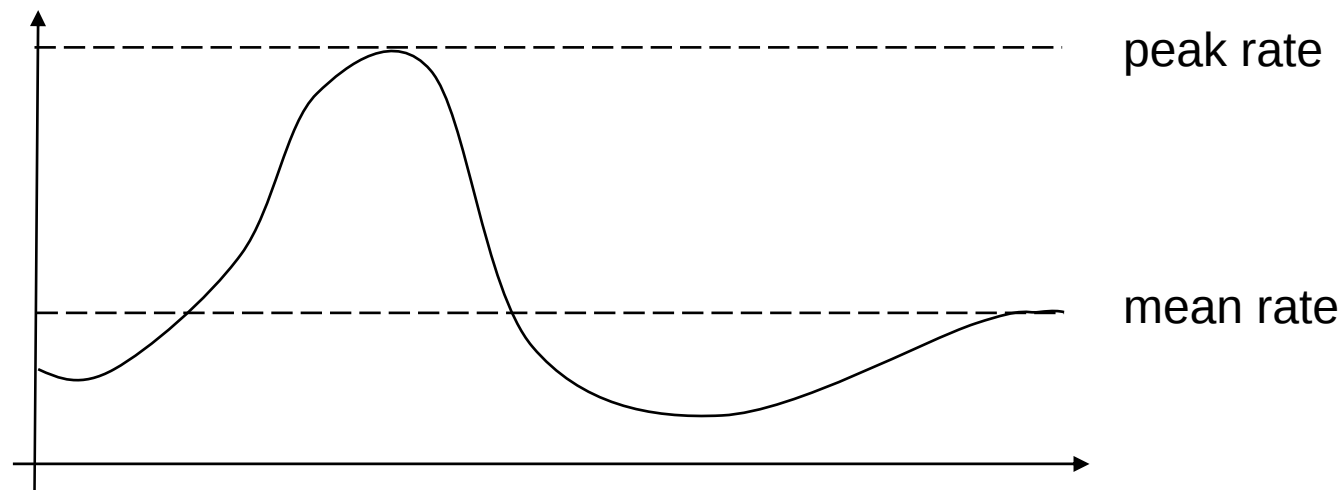


- Ez akkor jó megoldás, ha
 - fix adatráták vannak és a sávszélességet annak megfelelően osztjuk csatornákra
 - A források a vezetéket jól kihasználják



Löketszerűen érkező adatok (bursty traffic)

- Probléma: **bursty traffic**
 - Definíció: nagy különbség a forgalom **csúcs rátája** (**peak rate**) és az **átlagos rátája** (**mean** or **average rate**) között
 - Számítógép-hálózatokban $\text{peak rate} / \text{mean rate} = 1000/1$ nem szokatlan

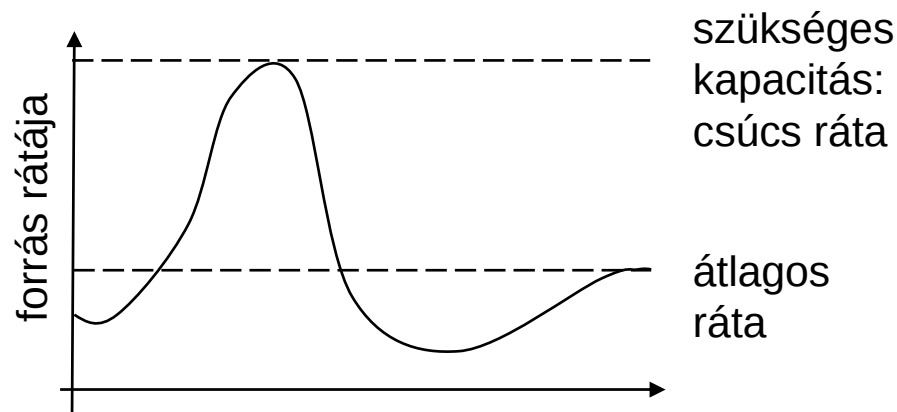


Löketszerűen érkező adatok és statikus multiplexálás

A linknek / csatornának statikus multiplexálás esetén vagy ... vagy ...

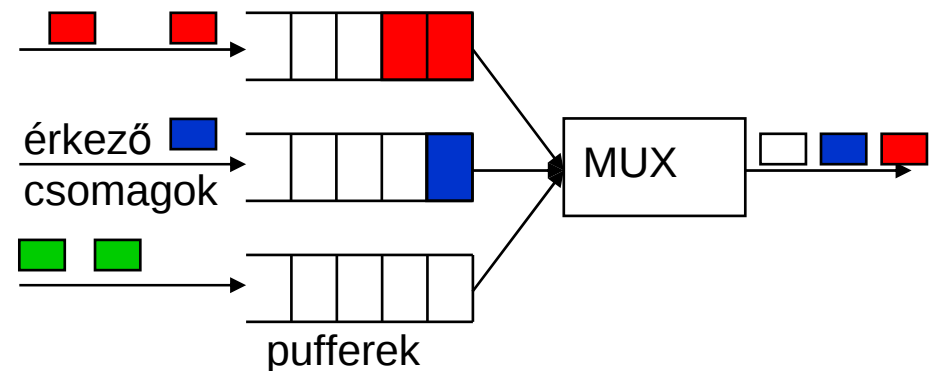
- vagy elegendően nagy kapacitásúnak kell lenni, hogy a csúcs rátát kezelni tudja

- Pazarlás, mert az átlagos ráta nem használja ki a csatornát



- vagy az átlagos rátára alapozva kell dimensionálni

- ekkor pufferek (queue) szükségesek
 - mi lesz a csomag késéssel (delay)?



-
- The diagram illustrates a packet scheduling process. On the left, three horizontal lines represent incoming packet streams. The top line is labeled 'érkező csomagok' (incoming packets) and shows two red packets. The middle line is labeled 'pufferek' (buffers) and shows four empty slots. The bottom line is labeled 'pufferek' and shows one green packet. Arrows indicate the flow of packets into the buffers. A central box labeled 'MUX' (multiplexer) receives input from the buffers. An arrow points from the MUX to the right, showing a sequence of packets: a red packet, an empty slot, another empty slot, and a final red packet.

MAC sub-réteg

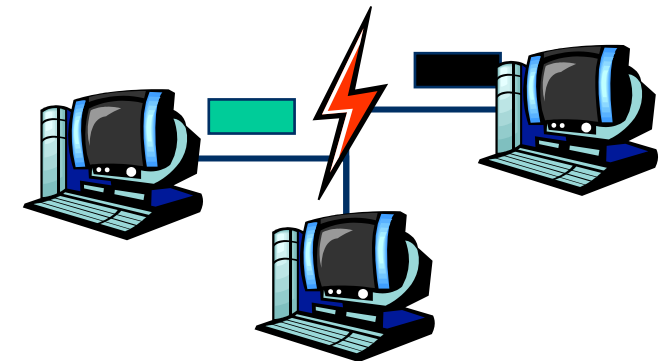
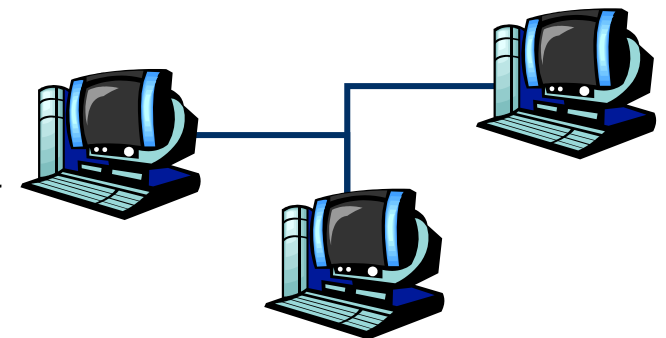
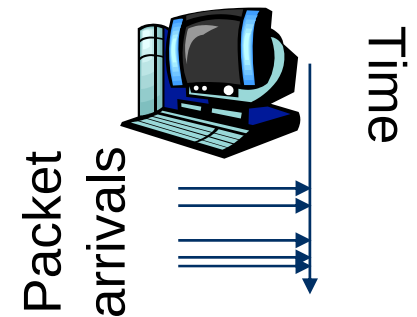
- Statikus Multiplexálás
- **Dinamikus csatorna foglalás**
 - Kollízió alapú protokollok
 - Verseny-mentes protokollok (contention-free)
 - Protokollok korlátozott versennyel (limited contention)
- Az Ethernet példája

Dinamikus csatorna foglalás – MAC

- Statikus multiplexálás nem megfelelő löketszerű adatforgalom kezelésére
 - Telefon hálózatok forgalma nem löketszerű, számítógépes hálózatoké az
- Alternatíva: A csatorna/link/erőforrás hozzárendelése ahhoz a forráshoz aki éppen adatot akar küldeni
 - Dinamikus csatorna foglalás (channel allocation)
 - az erőforrás fix részének hozzárendelése helyett
- Szabályozni kell a médium hozzáférést:
 - Médium hozzáférés protokoll (Medium Access Control protocol - MAC) szükséges

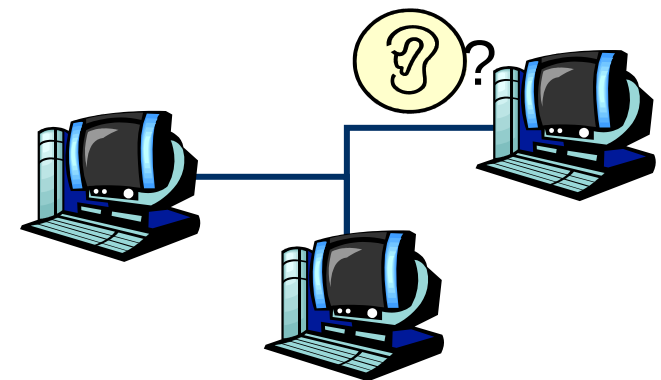
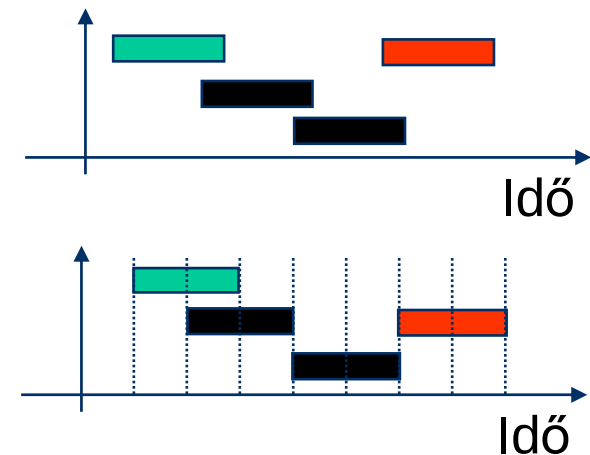
A dinamikus csatornafoglalás modellje

- **N állomás** (vagy N terminal)
 - N független állomás használja az adott erőforrást
 - Egy lehetséges **terhelés modell**: annak a valószínűsége, hogy egy állomás Δt intervallumban csomagot generál: $\lambda \Delta t$, ahol $\lambda = \text{konstans}$
- **Egy csatorna**
 - Az összes állomás részére együttesen egy csatorna áll rendelkezésre
 - A csatornán kívül semmilyen más lehetőség nincs kommunikálni egymással
- **Kollízió modell** (ütközés)
 - Egy időben csak egy frame vihető át eredményesen
 - Ha két (vagy több) frame időben átfedi egymást, akkor azok ütköznek és mindkettő szétrombolódik
 - Egy állomás se tudja fogadni egyik frame-et sem
 - Megjegyzés: ez alól a szabály alól van néha kivétel (pl. CDMA)



Modellek

- Időmodellek
 - Folytonos
 - Átvitel minden időben kezdődhet (nincs központi óra)
 - Diszkrét (Slotted time)
 - Az idő-tengely darabokra (slots) van osztva
 - Átvitel csak egy slot határán kezdődhet
 - Egy slot lehet üres (idle), vagy sikeresen átvitt, vagy kollíziót tartalmazó
- Vivő-érzékelés (Carrier Sensing)
 - Az állomások képesek felismerni, hogy éppen egy más állomás használja-e a csatornát
 - Nem feltétlenül megbízhatóan (pl. egy éppen kezdődő átvitelnél)
 - Ha a csatorna foglalt (busy), nem indít az állomás átvitelt



A hatékonyság mérése

- A csatornafoglalás hatékonyságának mértékei
- **Átvitel** (throughput)
 - Csomagok száma időegységenként
 - Különösen nagy terhelés esetén fontos
- **Késés** (delay)
 - Egy csomag átviteléhez szükséges idő
 - Alacsony terhelés esetén
- **Fairness**
 - Minden állomást egyenlőként kezelünk
 - Az átvitel és a késés körülbelül egyforma legyen az állomásokon

Átvitel és a feldolgozandó terhelés (offered load)

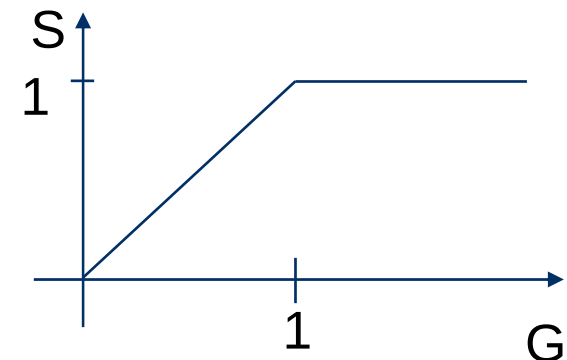
- **Feldolgozandó terhelés (offered load) G**

- A csomagok száma csomag-időegységenként, amit a protokollnak kezelnie kell

- $G > 1$: túlterhelés

- **Ideális protokoll**

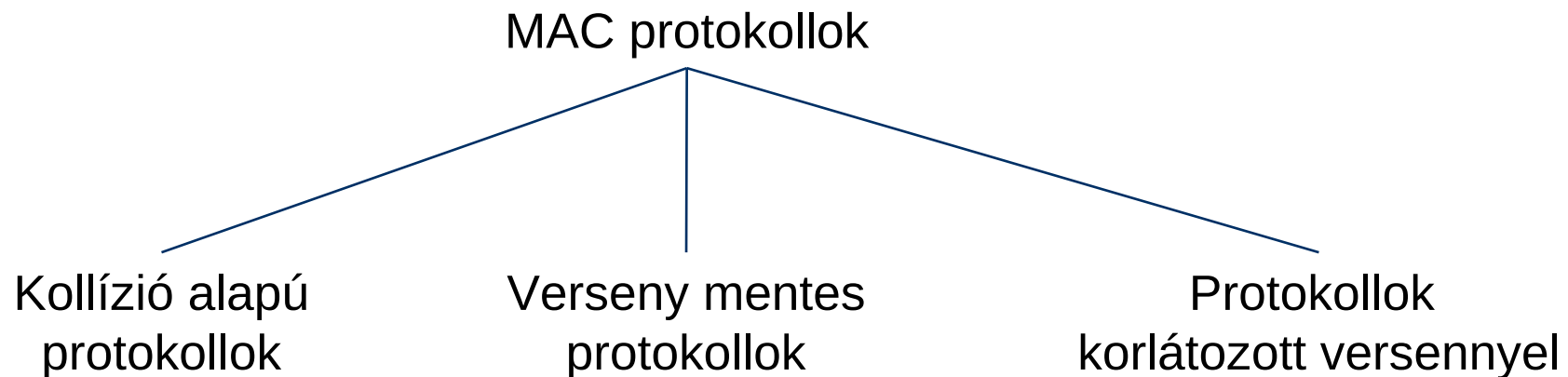
- Amíg $G < 1$, akkor az átvitel S egyenlő G -vel
- Ha $G \geq 1$, akkor $S = 1$



- És: konstans kis késés tetszőlegesen sok állomás esetén is

Lehetséges MAC-protokollok

- Fő megkülönböztetés: Megenged-e a protokoll kollíziót?
 - Rendszer döntés
 - A feltétlen kollízió-elkerülés a hatékonyság csökkenésével járhat



Rendszer, amelyben kollízió történhet:
Contention System (verseny rendszer)

MAC sub-réteg

- Statikus Multiplexálás
- Dinamikus csatorna foglalás
 - **Kollízió alapú protokollok**
 - Verseny-mentes protokollok (contention-free)
 - Protokollok korlátozott versennyel (limited contention)
- Az Ethernet példája

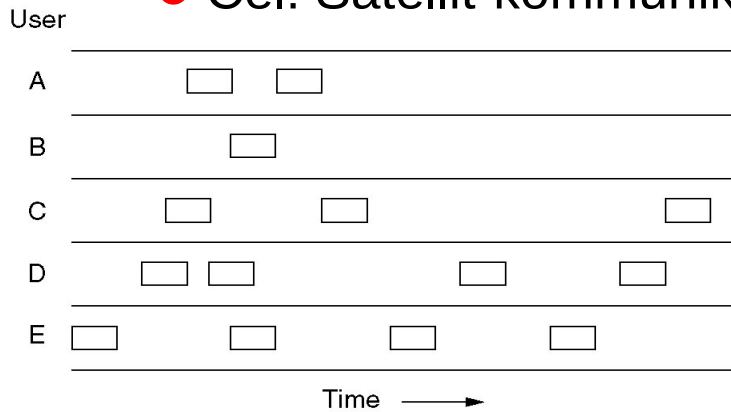
- **Algorithmus:**

- Amikor egy csomag kész, azonnal átvitelre kerül

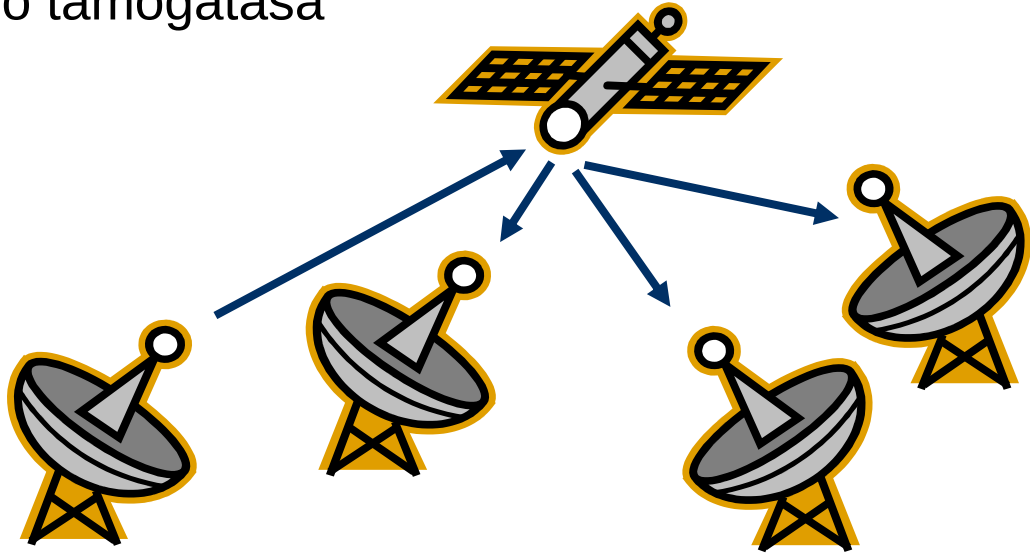
● Történet:

- 1985 by Abrahmson et al., University of Hawaii

- Cél: Satellit-kommunikáció támogatása



A csomagok tetszőleges időben
kerülnek átvitelre



ALOHA – Elemzés

- Előny
 - Egyszerű
 - Koordináció nem szükséges
- Hátrányok
 - Kollíziók
 - A küldő nem teszteli a csatorna állapotát
 - A küldőnek nincs direkt módszere arra, hogy megtudja, hogy eredményes volt-e az átvitel
 - Nyugták (ACK) szükségesek
 - A nyugták szintén ütközhetnek

ALOHA – Hatékonyság

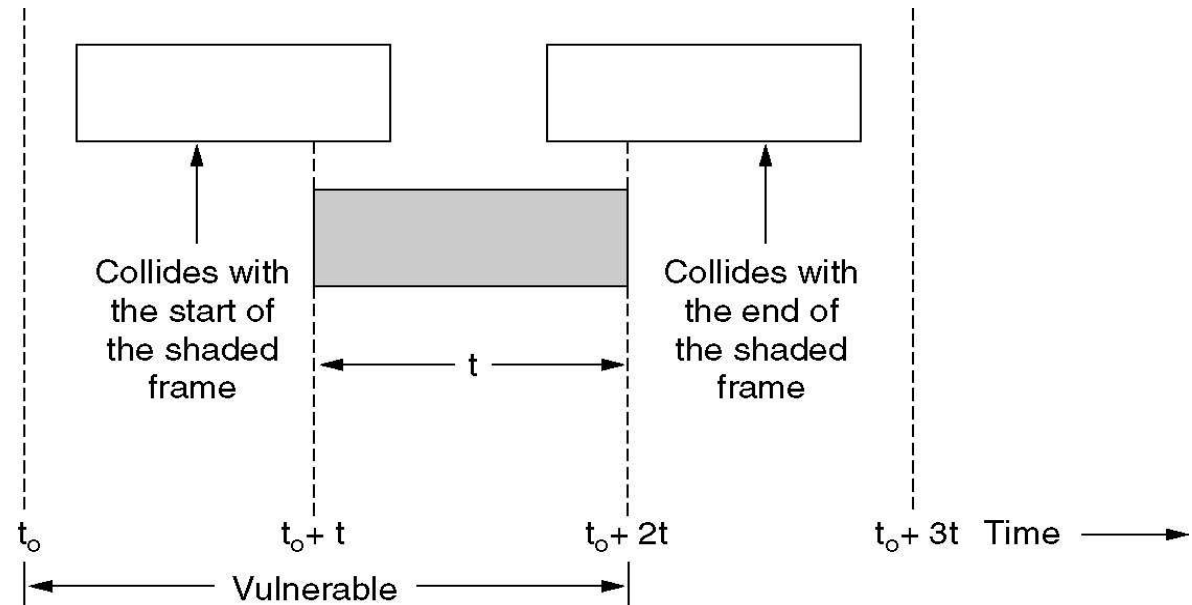
- Tegyük fel, hogy a csomagok létrehozása **Poisson-folyamat**:
 - „Végtelen” sok állomás, melyek egyformán, függetlenül viselkednek
 - Minden csomag egyforma hosszú, annak átviteléhez egységnyi idő kell
 - Az idő két küldési kísérlet között exponenciális eloszlású
 - Legyen G a küldési kísérletek számának várható értéke egységnyi idő alatt (egységnyi idő = egy csomag átviteléhez szükséges idő)
 - Ekkor:

$$P[k \text{ küldési kísérlet } t \text{ idő alatt}] = \frac{(Gt)^k}{k!} e^{-Gt}$$

- Ahhoz hogy sikeres átvitelt hajtsunk végre, nem szabad hogy kollízió lépjen fel egy másik csomaggal
- Mi ennek a valószínűsége?

ALOHA – Hatékonyság

- Egy X csomag ütközik, ha
 - egy csomag nem ért véget, amikor X indul
 - egy csomag kicsivel X vége előtt indul



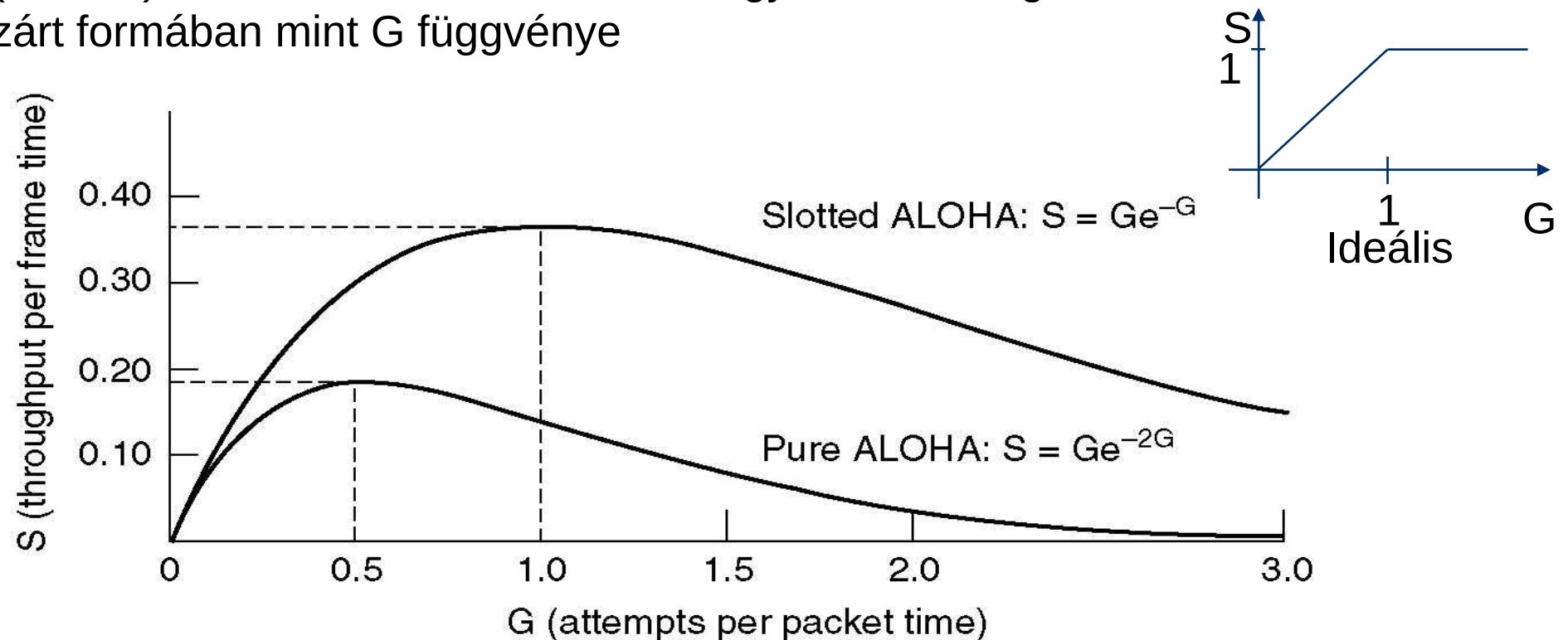
- Azaz, egy csomagátvitel akkor sikeres, ha két egységnyi időben nincs másik csomagátviteli kísérlet
 - Valószínűség: $P_1 = P(1 \text{ csomag } 2 \text{ egységnyi időben}) = 2G e^{-2G}$
 - Maximális átvitel $S(G) = (1 \text{ csomag} / 2 \text{ időegység}) * 2G e^{-2G}$
 - Optimum $G = 0,5$ -nál: $S = 1/(2e) \approx 0,184$

Egy javítás: Slotted ALOHA

- ALOHA problémája: a csomag „sebezhetőségi” ideje hosszú (2 időegység)
- Csökkentsük idődarabok (slot) bevezetésével – átvitel csak egy slot elején kezdődhet
 - Feltesszük, hogy a slot-ok szinkronizálása „valahogy” rendelkezésre áll
- Eredmény: Sebezhetőségi idő feleződik, az átvitel megduplázódik
 - $S(G) = Ge^{-G}$
 - Optimum $G=1$ -nél: $S=1/e$

Hatékonyság a feldolgozandó terhelés függvényében

- (Slotted) ALOHA esetén, az átvitel S egyszerűen megadható zárt formában mint G függvénye



→ Nem jó protokoll

- Az átvitel összezuhan, ha nő a terhelés!

Vivő-érzékelés (Carrier Sensing)

- (Slotted) ALOHA egyszerű, de nem kielégítő
- Stratégia: Figyeljünk mielőtt beszélünk (udvariasság segít)
- Figyeljük a vivő médiumot (carrier), hogy szabad-e, mielőtt adatot küldünk
 - **Carrier Sense Multiple Access (CSMA)**
 - Nem viszünk át adatot, ha nem szabad (egy másik állomás éppen adatot visz át)
- Alapvető kérdés: Hogyan viselkedjünk pontosan, ha a medium nem szabad?
 - Különösen: MIKOR próbáljuk újra az átvitelt?

1-persistent CSMA

- Ha a vivő médium nem szabad, várjunk, amíg szabad lesz
- Akkor azonnal kezdjük meg az átvitelt
- Ha kollíziót tapasztalunk, akkor
 - várjunk véletlenül választot ideig és ismételjük meg előlről
- „Türelmetlen” várakozás (persistent waiting)
- Nyilvánvaló probléma: ha több állomás vár, akkor *garantált* a kollízió!
- Azért jobb, mint az ALOHA vagy a slotted ALOHA

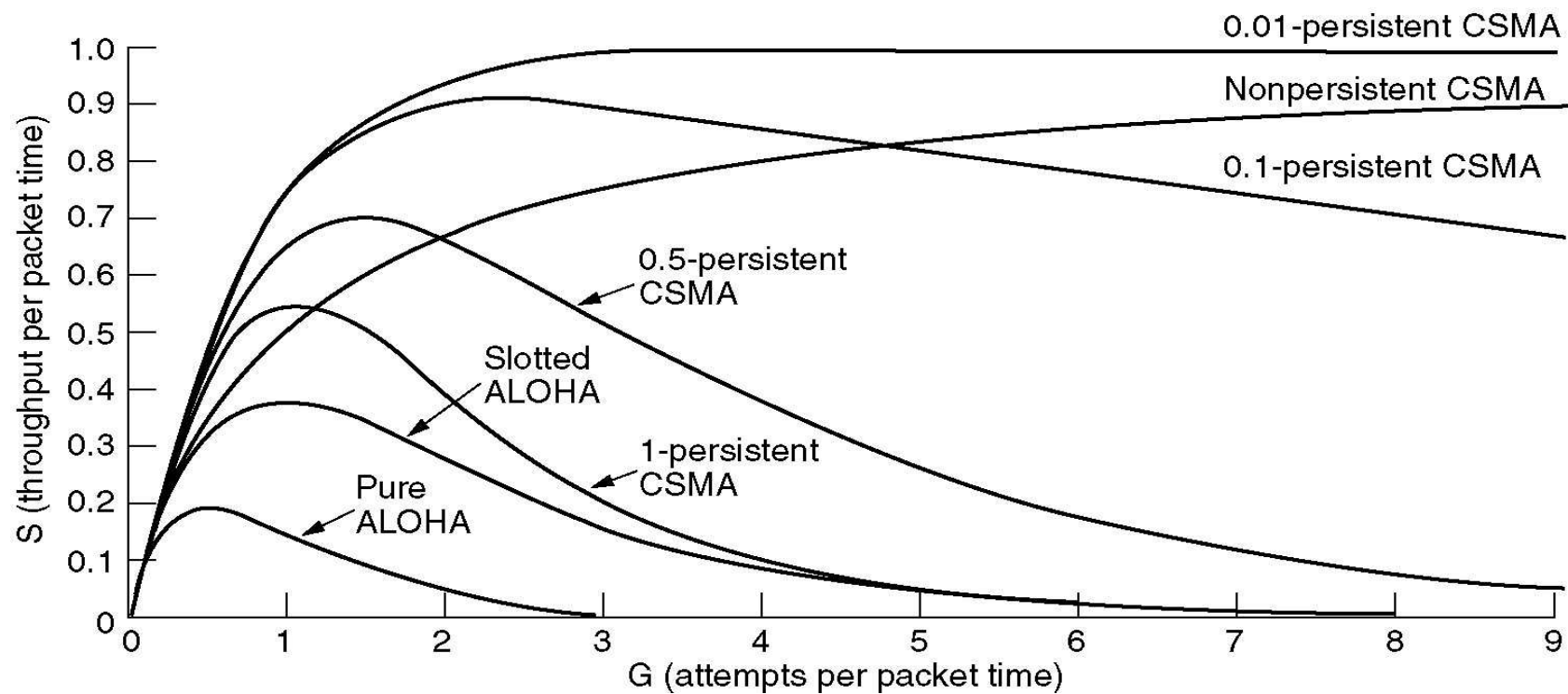
Non-persistent CSMA

- Ha a csatorna szabad, kezdjük meg az átvitelt
- Ha a csatorna nem szabad,
 - várjunk véletlenül választott ideig
 - utána ellenőrizzük újra, hogy a csatorna szabad-e, és így tovább
- A csatornát nem ellenőrizzük folyamatosan
 - kevésbé mohó
- A hatékonyság függ attól, hogy milyen eloszlás szerint választjuk a várakozási időt a következő ellenőrzésig
 - Általánosan, jobb átvitelt eredményez, mint a „persistent CSMA” magas terhelés esetén
 - Alacsony terhelés esetén a várakozás nem szükséges és pazarló

p-persistent CSMA

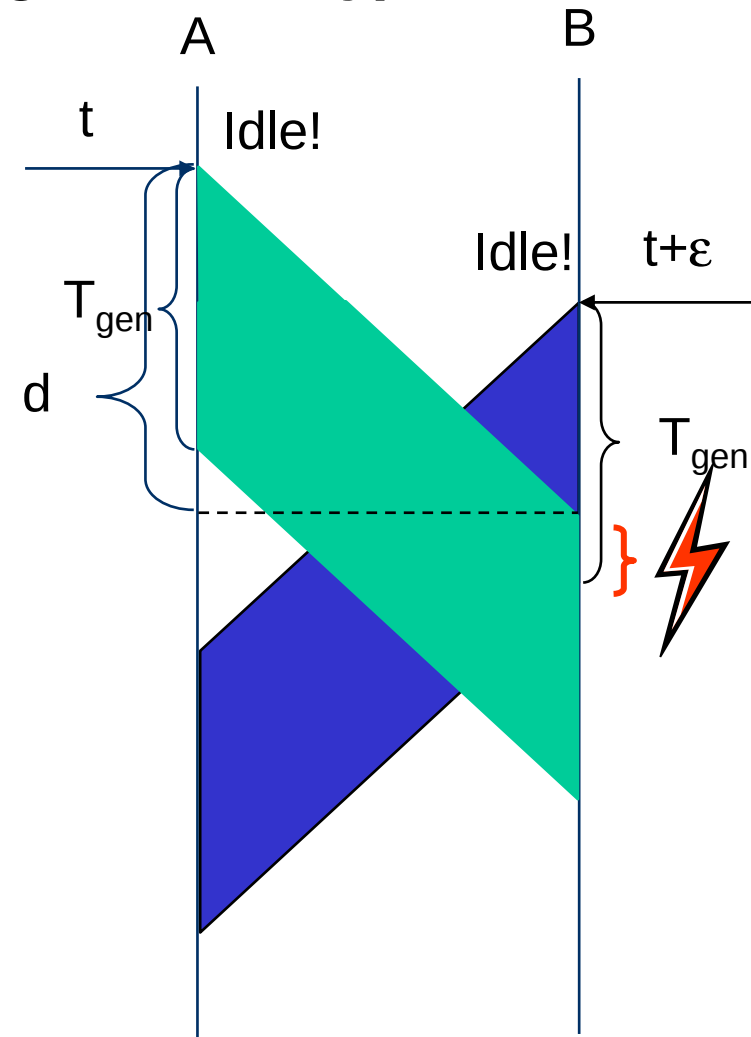
- A persistent és a non-persistent CSMA kombinációja
 - idő-slot modellt használ
- 1. Ha a csatorna szabad,
 - p valószínűséggel küldjük a csomagot
 - ha kollíziót tapasztalunk,
 - várjunk véletlen ideig
 - kezdjük újra az 1. pontban
 - egyébként ($1-p$ valószínűséggel)
 - várjunk a következő slot-ra
 - folytassuk az 1. pontban
- 2. Ha a csatorna foglalt
 - figyeljük folyamatosan, amíg nem lesz szabad, azután folytassuk az 1. pontban

CSMA hatékonysága



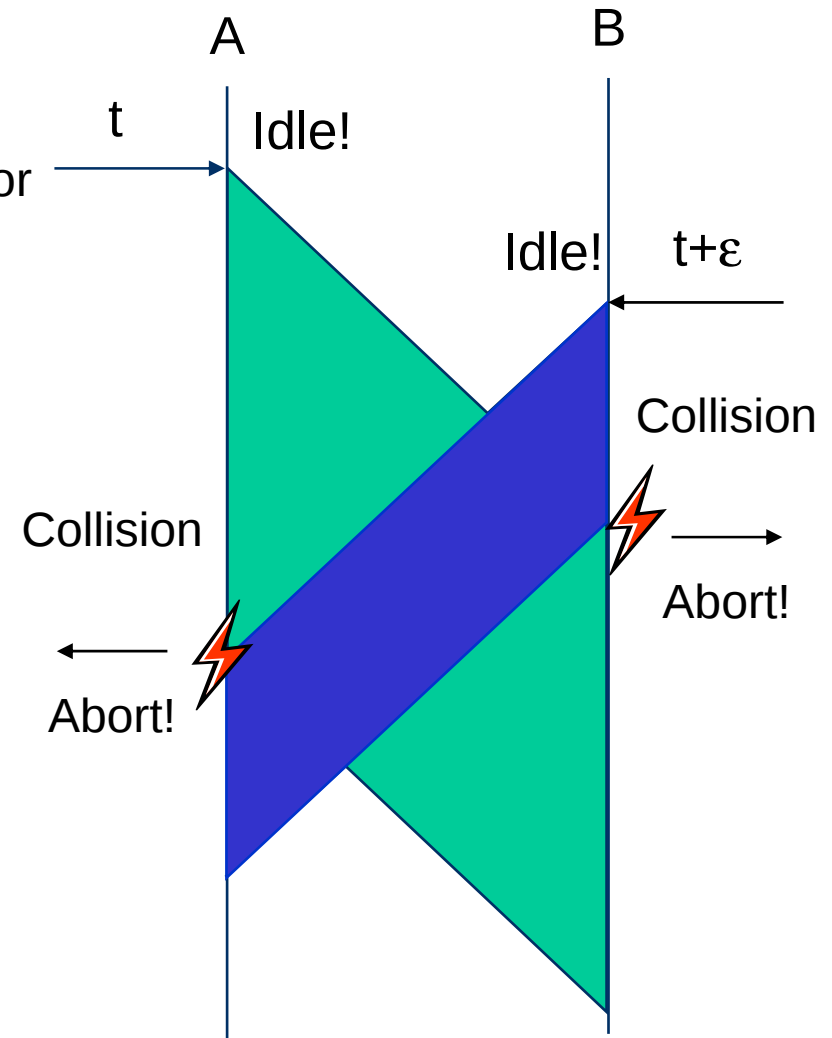
CSMA és propagációs késés (propagation delay)

- Minden CSMA sémának van egy elvi korlátja: A **propagációs késés d**
- Tegyük fel, két állomás lesz küldésre kész, az egyik t , a másik $t+\epsilon$ időpontban
 - t időpontban a csatorna teljesen szabad
 - Az állomások között a propagációs késés $d > \epsilon$
- A második állomás nem tudja érzékelni az első állomás már megkezdett átvitelét
 - Egy szabad csatornát érzékel, elindítja a küldést, és kollíziót okoz



Kollízió felismerés (collision detection) – CSMA/CD

- Ha két csomag ütközik, sok idő veszik el azok átvitelének befejezésére
 - Ha lehetséges lenne felismerni egy kollíziót amikor az fellép, az átvitelt lehetne abortálni és egy új próbát tenni
 - Az elvesztegetett idő csökken, nem kell megvárni, hogy a (szétrombolt) csomagok befejeződjenek
 - A fizikai rétegtől függően, a kollízió felismerhető!
 - Szükséges: A küldőnek képesnek kell lenni „hallgatni” a médiumot miközben küld és összehasonlítani amit küld és amit „hall”
 - Ha különbözik: Kollízió
- **CSMA/CD – Carrier Sense Multiple Access/Collision Detection**
- Feltétel, hogy felismerjük mindkét oldalon:
$$T_{\text{gen}} \geq 2d$$
 - T_{gen} : csomag generálási ideje

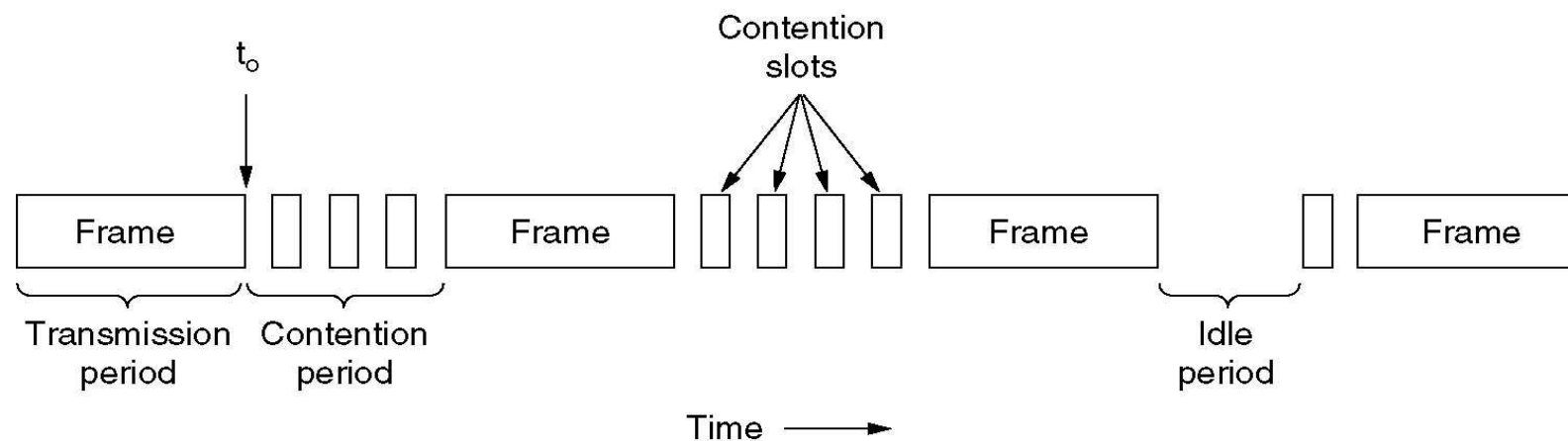


Mi a teendő kollízió esetén?

- Az állomások át akarják vinni a csomagjaikat a kollízió ellenére
 - Újra meg kell próbálniuk
 - Azonnal? Ez egy másik kollíziót okozna
 - Valahogy koordinálva? Nehéz, nem áll rendelkezésre kommunikációs médium
 - Várjunk egy véletlen ideig!
 - Randomizálás “de-szinkronizálja” a médium hozzáférést, és ezzel segít elkerülni a kollíziót
 - Valamennyi kihasználatlan időt eredményez
- Váltakozva verseny- és átviteli-periódusok

CSMA/CD periódusai

- Üres periódus (IDLE)
 - Egyik állomás sem küld frame-et
 - Verseny periódus (Contention Period)
 - Kollíziók történhetnek, az átvitel abortálódik
 - Átviteli periódus (Transmission Period)
 - Nincs Kollízió, a protokoll effektív része
- Csak verseny-, átviteli- és üres periódus van



Hogy válasszuk meg a véletlen várakozási időt?

- A legegyszerűbb választás: Válasszunk ki egyet k slot közül
 - Egyszerűség kedvéért tételezzünk fel egy slot-okra osztott idő modellt
 - Egyenletes eloszlás szerint $\{0, \dots, k-1\}$ felett
 $[0, \dots, k-1]$: verseny ablak (**contention window**)
 - Kérdés: hogy válasszunk meg k -t?
 - Kicsi k : Kicsi delay, de nagy az esély ismételt kollízióra
 - Nagy k : Kicsi az ismételt kollízió esélye (mivel az állomások kísérletei egy nagy intervallumra oszlanak el), de szükségtelenül nagy a delay, ha csak kevés állomás akarja használni a csatornát
- **Adaptáljuk** k választásához az állomások aktuális számát / csatorna terhelést

Hogyan változtassuk k -t a terheléstől függően?

- Egy lehetőség: derítsük ki *valahogy* explicit az állomások számát, számítsunk ki ehhez egy optimális k -t, tudassuk ezt minden állomással
 - Nehéz, magas overhead, ...
 - Lehetséges egy *implicit* megoldás?
- Milyen következményekkel jár egy kicsi k , ha a terhelés nagy?
 - Sok kollízió!
 - Tehát: Használjuk a kollíziókat indikátorként, hogy a verseny ablak túl kicsi – növeljük meg a verseny ablak méretét!
 - Csökkenteni a kollíziók valószínűségét, automatikusan adaptálja a terhelés növekedését
- Kérdés: Hogy növeljük k -t a kollízió után, hogy csökkentsük újra?

Hogy változtassuk k-t – Binary exponential backoff

- Növeljük k-t a **kollízió után**: sok lehetőség van
 - Általánosan használt: **duplázzuk** meg k-t
 - De csak egy korlátig, mondjuk, 1024 slot – kezdjük k=2-vel
 - Ezt a stratégiát **binary exponential backoff**-nak hívják
- Csökkentsük k-t, ha elegendően sok frame kollízió mentesen átvitelre került
 - Lehetőségek: vonjunk ki belőle egy konstans (TCP), felezzük meg, ...
 - Viszonylag komplikált, erőforrást pazarolhat, miközben nem elég agilis
 - A legegyszerűbb: induljunk megint k=1-gyel
 - Általánosan használt

Hogy változtassuk k-t – Binary exponential backoff

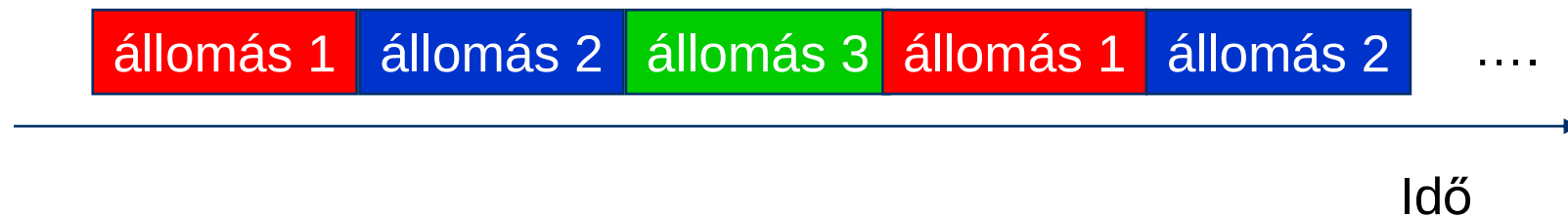
- Algoritmus **binary exponential backoff**
 - $k := 2$
 - Amíg az utolsó küldésnél kollízió történt
 - Válasszuk i -t egyenlő valószínűséggel véletlenül $\{0, \dots, k-1\}$ közül
 - Várjunk i slot-ot
 - Küldjük a frame-et (kollízió felismerése esetén: abort)
 - Ha $k < \text{limit}$: $k := 2k$
- Ez az algoritmus
 - a várakozási időt dinamikusan a csatornát használó állomások számához igazítja
 - gondoskodik a csatorna egyenletes kihasználásáról
 - fair (hosszú távon)

MAC sub-réteg

- Statikus Multiplexálás
- Dinamikus csatorna foglalás
 - Kollízió alapú protokollok
 - **Verseny-mentes protokollok (contention-free)**
 - Protokollok korlátozott versennyel (limited contention)
- Az Ethernet példája

Verseny mentes protokollok

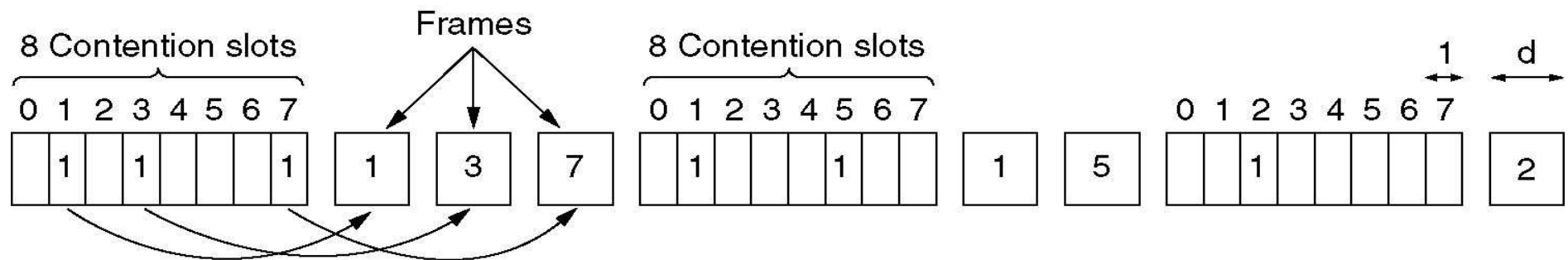
- Egyszerű példa: Statikus (idő-) multiplexálás (TDMA)
 - Minden állomáshoz egy fix idő-slotot rendelünk egy ismétlődő idő idő-séma szerint



- Hátrányait elemeztük
- Van-e dinamikus kollízió mentes protokoll?

Bit-map protokoll

- A TDMA problémája
 - Ha az állomás nem küld semmit, az idő-slotja kihasználatlan
- Foglálási rendszer: Bit-map protocol
 - Rövid statikus foglalás-slotok, melyek jelzik az átvitel kívánságot
 - Minden állomásnak hallani kell



Bitmap-Protokollok

- Tulajdonságok alacsony terhelés esetén
 - Ha nincs csomagküldés, akkor az (üres) verseny-slot ismétlődik
 - Egy állomás, ha küldeni akar, meg kell várnia a verseny-slotokat
 - Viszonylag nagy késés (delay)
- Tulajdonságok nagy terhelés esetén
 - A csatornát az adatcsomagok dominálják
 - Az adatcsomagok nagyobbak mint a verseny-slotok
 - Az overhead elhanyagolható
 - Jó és stabil átvitel (throughput)
- Bitmap egy Carrier-Sense protokoll!