Számítógépes Hálózatok és Internet Eszközök

2007

11. Adatkapcsolati réteg, MAC alréteg –
CSMA, versenymentes protokollok, korlátozott verseny

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 elölről
- „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

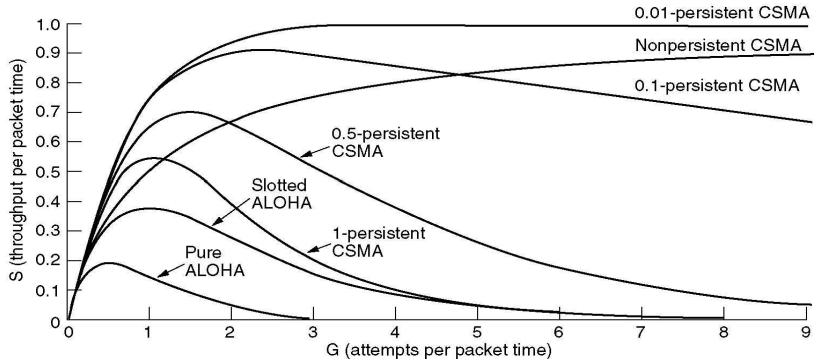
- Ha a csatorna szabad, kezdjük meg az átvitelt
- Ha a csatorna nem szabad,
  - várjunk véletlenül választott ideig
  - utána ellenőrizzük újra, hogy a csatorna szabad-e, és így tovább
- A csatornát nem ellenőrizzük folyamatosan
  - kevésbé mohó
- A hatékonyság függ attól, hogy milyen eloszlás szerint 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

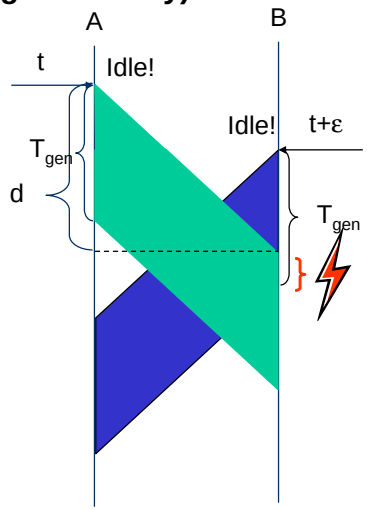
1. Ha a csatorna szabad,
   - p valószínűséggel küldjük a csomagot
     - ha kollíziót tapasztalunk,
       - várjunk véletlen ideig
       - kezdjük újra az 1. pontban
   - egyébként (1-p valószínűséggel)
     - várjunk a következő slot-ra
     - folytassuk az 1. pontban
2. Ha a csatorna foglalt
   - 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+ε időpontban
  - t időpontban a csatorna teljesen szabad
  - Az állomások között a propagációs késés d > ε
- A második állomás nem tudja érzékelni az első állomás már megkezdett átvitelét
  - Egy szabad csatornát érzékel, elindítja a küldést, és kollíziót okoz

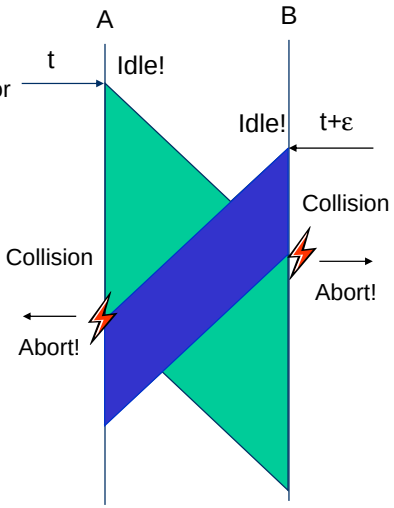


Kollízió felismerés (collision detection) – CSMA/CD

- 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:
  $T_{gen} \geq 2d$
  - T_{gen}: csomag generálási ideje



Mi a teendő kollízió esetén?

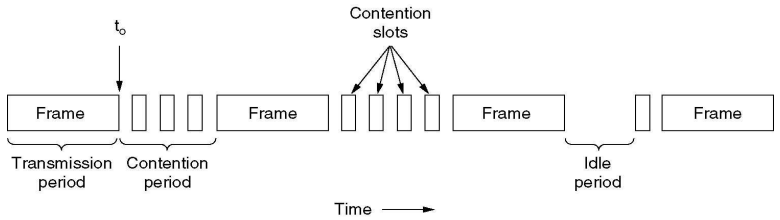
- Az állomások át akarják vinni a csomagjaikat a kollízió ellenére
- Újra meg kell próbálniuk
  - Azonnal? Ez egy másik kollíziót okozna
  - Valahogy koordinálva? Nehéz, nem áll rendelkezésre kommunikációs médium
  - Várjunk egy véletlen ideig!
    - Randomizálás “de-szinkronizálja” a médium hozzáférést, 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,…, k-1} felett
    [0,…, 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 kisérletei 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!
    - Csökkeni 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 konstanst, 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

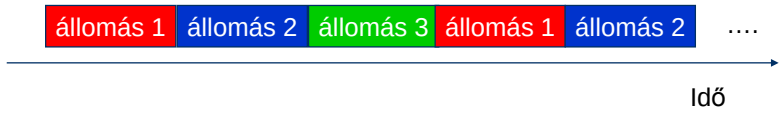
- Algoritmus **binary exponential backoff**
  - k := 2
  - Amíg az utolsó küldésnél kollízió történt
    - Válasszuk i-t egyenlő valószínűséggel véletlenül {0,...,k-1} közül
    - Várjunk i slot-ot
    - Küldjük a frame-et (kollízió felismerése esetén: abort)
    - Ha k < limit: k := 2 k
- Ez az algoritmus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

MAC alréteg

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

Versenymentes protokollok

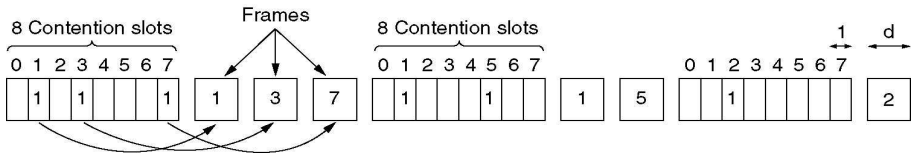
- Egyszerű példa: Statikus (idő-) multiplexálás (TDMA)
  - Minden állomáshoz egy fix idő-slotot rendelünk egy ismétlődő idő idő-séma szerint



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

Bit-map protokoll

- A TDMA probémája
  - Ha az állomás nem küld semmit, az idő-slotja kihasználatlan
- Foglalási 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 (througput)
- Bitmap egy Carrier-Sense protokoll!

MAC alréteg

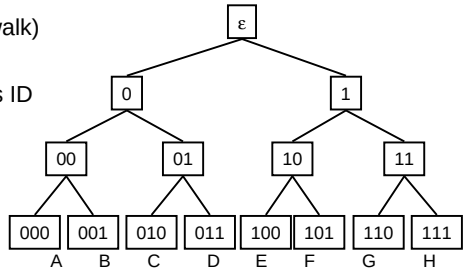
- Statikus Multiplexálás
- Dinamikus csatorna foglalás
  - Kollízió alapú protokollok
  - Verseny-mentes protokollok (contention-free)
  - **Protokollok korlátozott versennyel (limited contention)**
- Az Ethernet példája
- WiFi példája

Protokollok korlátozott versennyel

- Cél:
  - kis késés (delay) alacsony terhelés esetén
    - mint a kollízió alapú protokolloknál
  - nagy átvitel (throughput) nagy terhelés esetén
    - mint a verseny mentes protokolloknál
- → korlátozott verseny (verseny a verseny slotoknál)
- Ötlet:
  - A verseny slotokhoz vegyük figyelembe a résztvevő állomások számát
  - Több állomásnak kell használni egy slotot

Adaptív fa bejárás protokoll

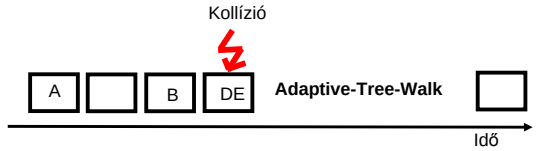
- Adaptív fa bejárás protokoll (adaptive tree walk)
- Kiindulópont:
  - Minden állomást egy egyértelmű, bináris ID reprezentál
  - Az ID-k egy fa leveleinek felelnek meg
  - Szinkronizált protokoll
  - A fa egy u csomópontjánál 3 esetet különböztethetünk meg:
    - nem küld állomás u részfájában
    - pontosan egy állomás küld
    - kollízió: legalább két állomás küld



Állomások

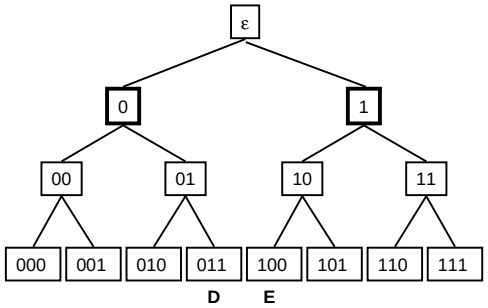
Adaptív fa bejárás protokoll – Alap algoritmus

- Alap_Algoritmus
  - Minden állomás azonnal küld (slotted Aloha)
  - Ha kollízió lép fel,
    - Egy állomás sem fogad el új csomagot a hálózati rétegtől
    - Hajtsuk végre az **adaptive_tree_walk(ε)** eljárást

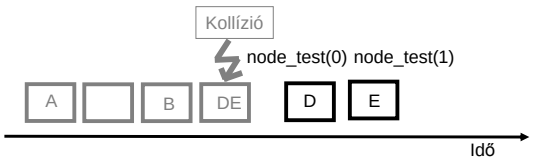


Adaptív fa bejárás protokoll – Csomópont teszt

- Csomópont-teszt algoritmus (node_test)
  - a fa egy u csomópontjához
- **node_test(u)**
  - Tekintsünk egy slotot u-hoz
  - A slotban azon állomások küldenek, amelyek u részfájában vannak (amelyek ID-ja u-val kezdődik)

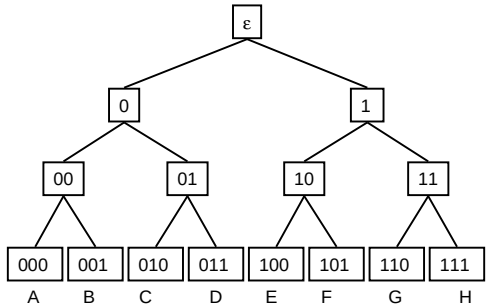


Kollíziót okozó állomások



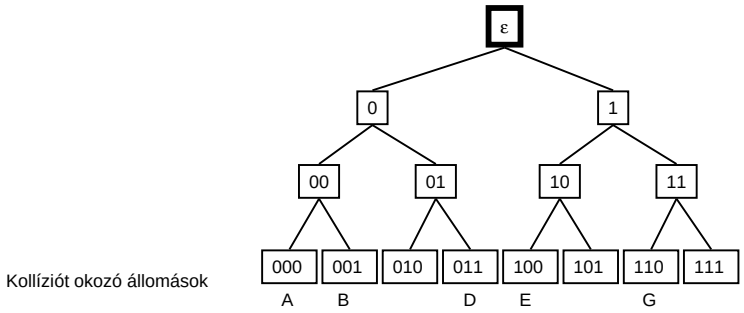
Adaptív fa bejárás protokoll – Alap algoritmus

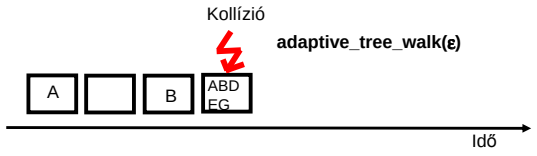
- Csomópont-teszt algoritmus
  - a fa egy u csomópontjához
- **node_test(u)**
  - Tekintsünk egy slotot a fa u csomópontjához
  - A slotban azon állomások küldenek, amelyek u részfájában vannak (amelyek ID-je u-val kezdődik)
- **adaptive_tree_walk(x)**
  - node_test(x0)
  - Ha kollízió lép fel,
    - adaptive_tree_walk(x0)
  - node_test(x1)
  - Ha kollízió lép fel,
    - adaptive_tree_walk(x1)



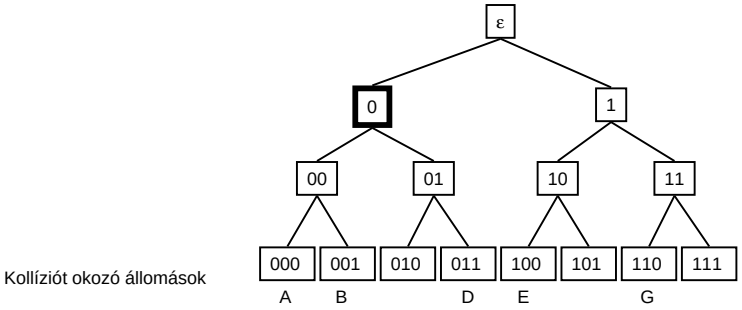
állomások

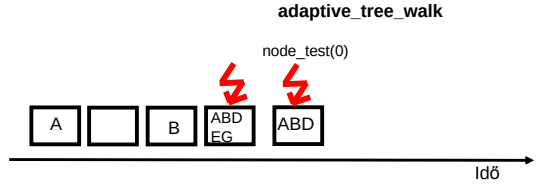
Adaptív fa bejárás protokoll – Példa (1)



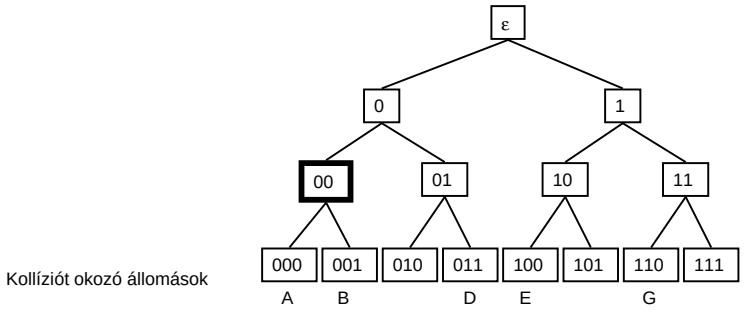


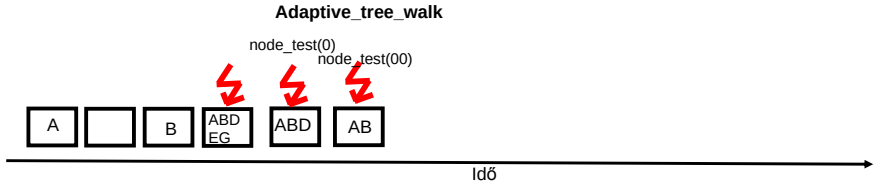
Adaptív fa bejárás protokoll – Példa (2)



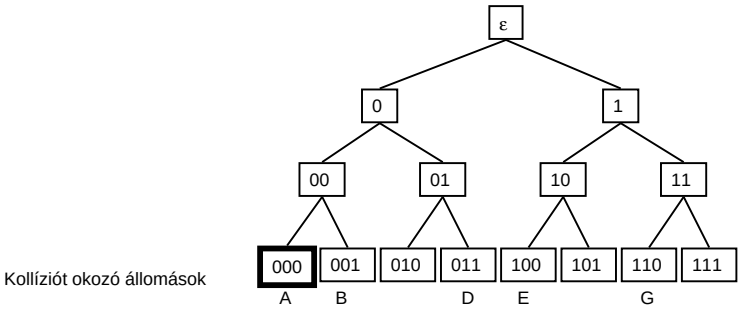


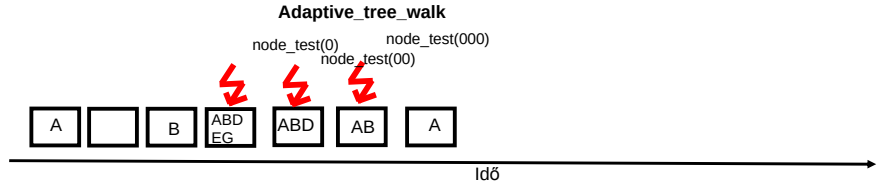
Adaptív fa bejárás protokoll – Példa (3)



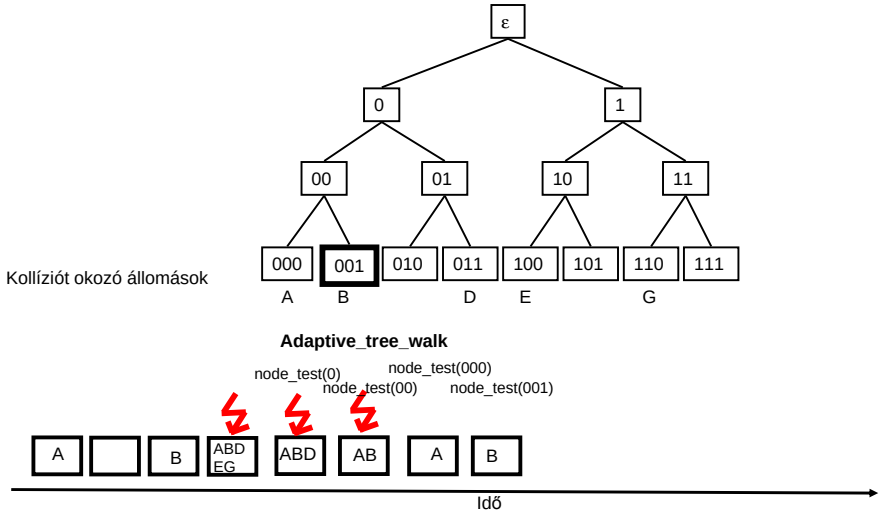


Adaptív fa bejárás protokoll – Példa (4)

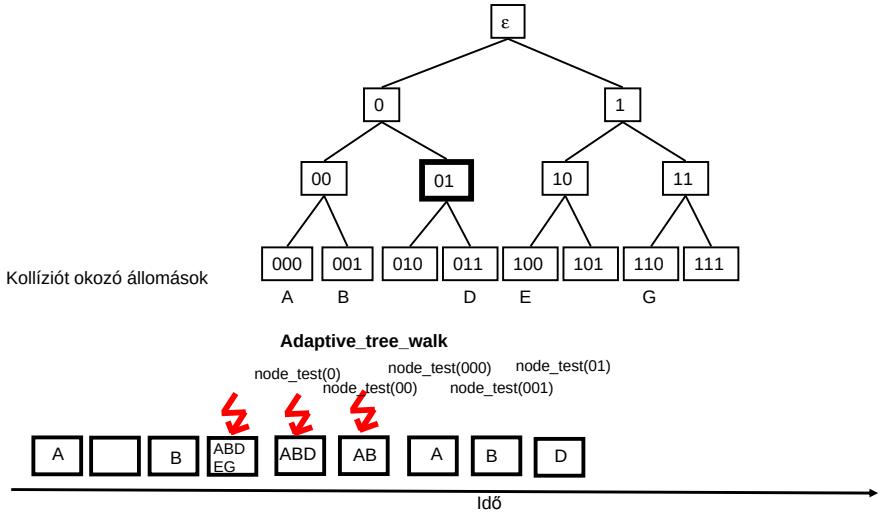




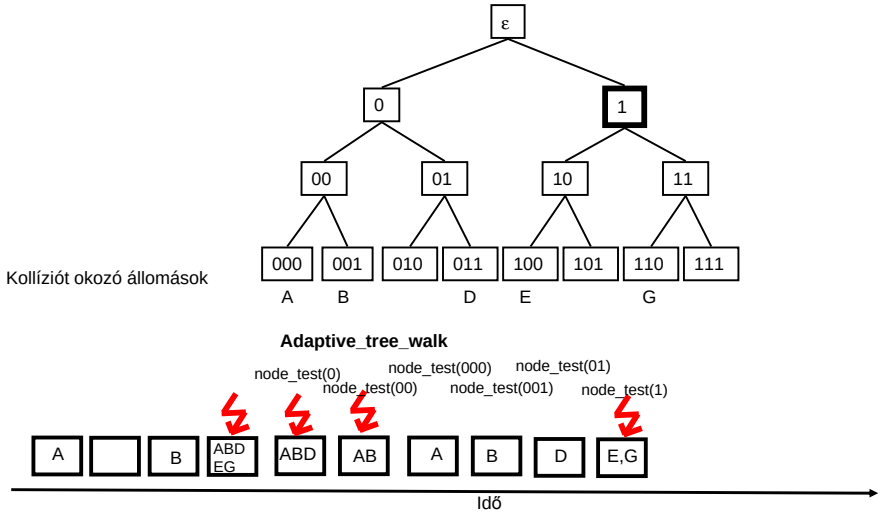
Adaptív fa bejárás protokoll – Példa (5)



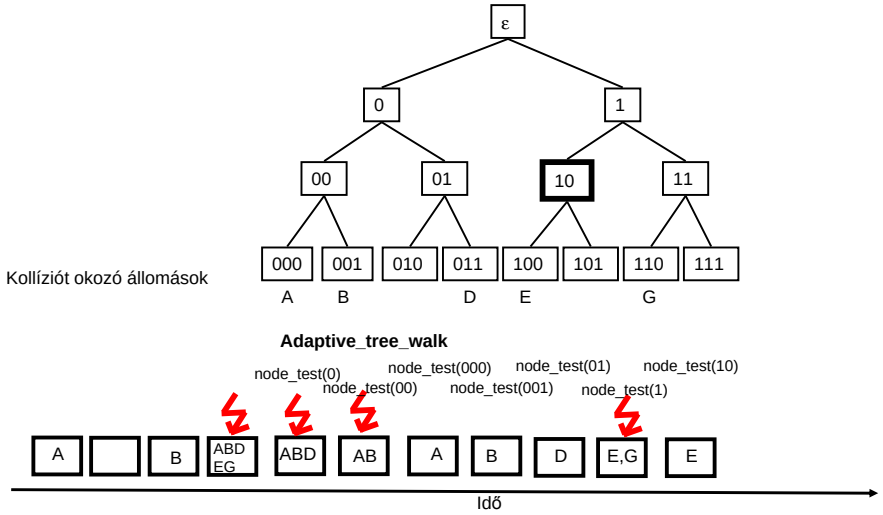
Adaptív fa bejárás protokoll – Példa (6)



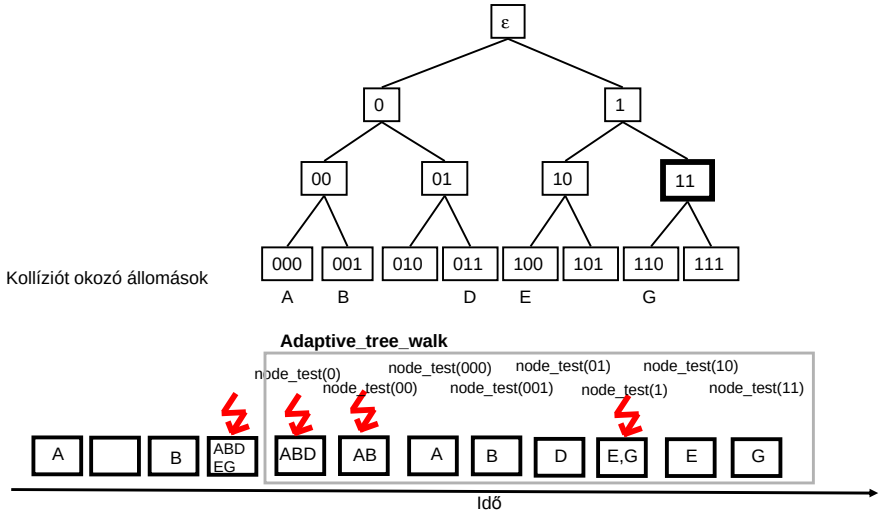
Adaptív fa bejárás protokoll – Példa (7)



Adaptív fa bejárás protokoll – Példa (8)



Adaptív fa bejárás protokoll – Példa (9)



Verseny mentes protokollok a vezeték nélküli kommunikációban