Algorithms

CRC

G(x) egy generátor polinom, aminek a foka r, ezt a polinomot a küldő és a vevő egyaránt ismeri.

Algoritmus

1. Fűzzünk r darab 0 bitet a keret alacsony helyiértékű végéhez, tehát az m + r bitet fog tartalmazni és reprezentálja a xr * M(x) polinomot
2. xr * M(x) elosztása G(x)-szel modulo 2 (bitsorozatok)
3. Az előző osztás maradékának kivonása xr * M(x)-ből modulo 2 (bitsorozatok), az eredmény az ellenőrző összeggel ellátott, továbbítandó keret. Jelölése: T(x)
4. A vevő a T(x) + E(x) polinomnak megfelelő sorozatot kapja, ahol E(x) a hibapolinom. Ezt elosztja a G(x) generátor polinommal, ha van maradéka ennek az osztásnak, akkor hiba történt.

CDMA

A kódosztásos többszörös hozzáférés (angolul Code Division Multiple Access, röviden CDMA) a multiplexálás egy formája és a többszörös hozzáférés egy lehetséges megvalósítása, amely az adatokhoz csatornánként speciális kódokat rendel, és kihasználja a konstruktív interferencia tulajdonságot a multiplexáláshoz.

Algoritmus

1. Minden bitidőt m darab rövid intervallumra osztunk, ezek a töredékek (angolul chip)
2. Minden állomáshoz egy m bites kód tartozik, úgynevezett töredéksorozat (angolul chip sequence)
3. Ha 1-es bitet akar továbbítani egy állomás, akkor elküldi a saját töredéksorozatát
4. Ha 0-es bitet akar továbbítani egy állomás, akkor elküldi a saját töredéksorozatának egyes komplemensét
5. m-szeres sávszélesség válik szükségessé, azaz szórt spektrumú kommunikációt valósít meg
6. Szemléltetésre bipoláris kódolást használunk:

   Bináris 0 esetén -1, bináris 1 esetén +1
   az állomásokhoz rendelt töredék sorozatok páronként ortogonálisak

CSMA

Az 1-perzisztens CSMA protokoll:

- Működése:
  - Vivőjelérzékelés van, azaz minden állomás belehallgathat a csatornába.
  - Folytonos időmodellt használ a protokoll.

  - Keret leadása előtt belehallgat a csatornába:
    1. Ha foglalt, akkor addig vár, amíg fel nem szabadul. Szabad csatorna esetén azonnal küld. (perzisztens)
    2. Ha szabad, akkor küld.
  - Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.
- Tulajdonságok:
  - A terjedési késleltetés nagymértékben befolyásolhatja a teljesítményét.
  - Jobb teljesítményt mutat, mint az ALOHA protokollok.

A nem-perzisztens CSMA protokoll:

- Működése:
  - Vivőjelérzékelés van, azaz minden állomás belehallgathat a csatornába.
  - Folytonos időmodellt használ a protokoll.
  - Mohóság kerülése.
  - Keret leadása előtt belehallgat a csatornába:
    1. Ha foglalt, akkor véletlen ideig vár (nem figyeli a forgalmat), majd kezdi előről a küldési algoritmust. (nem-perzisztens)
    2. Ha szabad, akkor küld.
  - Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.
- Tulajdonságok:
  - Jobb teljesítményt mutat, mint az 1-perzisztens CSMA protokoll. (intuitív)

A p-perzisztens CSMA protokoll:

- Működése:
  - Vivőjel érzékelés van, azaz minden állomás belehallgathat a csatornába.
  - Diszkrét időmodellt használ a protokoll.
  - Adás kész állapotban az állomás belehallgat a csatornába:
    1. Ha foglalt, akkor vár a következő időrésig, majd megismétli az algoritmust.
    2. Ha szabad, akkor p valószínűséggel küld, illetve 1-p valószínűséggel visszalép a szándékától a következő időrésig. Várakozás esetén a következő időrésben megismétli az algoritmust.Ez addig folytatódik, amíg el nem küldi a keretet, vagy amíg egy másik állomás el nem kezd küldeni, mert ilyenkor úgy viselkedik, mintha ütközés történt volna.
  - Ha ütközés történik, akkor az állomás véletlen hosszú ideig vár, majd újrakezdi a keret leadását.

A CSMA/CD protokoll:

- (CD → Collision Detection: ütközés érzékelés) Ütközés érzékelés esetén meg lehessen szakítani az adást.(„Collision Detection”):
- Működése:

- Minden állomás küldés közben is figyeli a csatornát,
- Ha ütközést tapasztal, azonnal megszakítja az adást (nem adja le a teljes keretet), véletlen ideig vár,majd újra elkezdi leadni a keretét.
- Az ütközés detektálás minimális ideje az az idő, ami egy jelnek a két legtávolabbi állomás közötti átviteléhez szükséges.
- Egy állomás megszerezte a csatornát, ha minden más állomás érzékeli az átvitelét.
- Az ütközés detektálás működéséhez szükséges a keretek hosszára egy alsó korlátot adnunk
- Ethernet a CSMA/CD-t használja
- Alapvetés: a közeg lehetőséget ad a csatornába hallgatásra
- Gyér forgalom esetén a közeghozzáférés nagyon gyors, mivel kevés állomás kíván a csatornán adni. Nagy hálózati forgalom esetén az átvitel lelassul, mivel a nagy csatorna terhelés miatt gyakoriak lesznek az ütközések. (A széles körben elterjedt Ethernet hálózat ezt a módszert használja.)

Algoritmus

1. Használjuk valamely CSMA variánst
2. A keret kiküldése után, figyeljük a közeget, hogy történik-e ütközés
3. Ha nem volt ütközés, akkor a keretet leszállítottuk
4. Ha ütközés történt, akkor azonnal megszakítjuk a küldést. Miért is folytatnánk hisz a keret már sérült…
5. Alkalmazzuk az bináris exponenciális hátralék módszert az újraküldés során (binary exponential backoff)

Ütközések történhetnek, az ütközéseket gyorsan észleljük és felfüggesztjük az átvitelt.

ALOHA:

Egyszerű ALOHA protokoll

A csatornakiosztás problémáját oldja meg. A rendszer lényege hogy a felhasználó bármikor adhat, ha van továbbítandó adata. De ha bárki bármikor adhat, akkor valószínű, hogy ütközések lesznek. A küldő azonban figyelheti a csatornát, így meg tudja állapítani hogy a keret tönkrement-e vagy sem. Ütközés esetén véletlen ideig vár az újraküldéssel.

Tulajdonságok:

- ALOHA protokollok áteresztő képessége egyforma keretméret esetén maximális.
- Keret idő – egy szabványos, fix hosszúságú keret átviteléhez szükséges idő
- Tegyük fel, hogy a felhasználók végtelen populációja a kereteket Poisson-eloszlás szerint állítja elő.
- Keretidőnként átlagosan N-et, ha:
  - N>1, akkor a csatorna túlterhelt.
  - 0<N≤1, akkor a csatorna áteresztő képessége elfogadható.

  - Tegyük még fel, hogy keretidőnként k számú új és régi keret együttes elküldési kísérleteinek valószínűsége
- ugyancsak Poisson-eloszlású, és keretidőnkénti középértéke G, ha
  - G=N, akkor a terhelés kicsi.
  - G>N, akkor a terhelés nagy.
  - Áteresztő képesség: S = □□0, ahol P0 keret sérülésmentes átvitelének valószínűsége.

Réselt ALOHA protokoll

Az idő diszkrét, keretidőhöz igazodó időszeletekre osztásával az ALOHA rendszer kapacitása megduplázható. (1972, Roberts) Következmény:

- kritikus szakasz hossza a felére csökken, azaz: □0 = □(−□)
- az áteresztő képesség: S = □□0 = □□(−□)
- A csatorna terhelésének kis növekedése is drasztikusan csökkentheti a médium teljesítményét.

Távolságvektor alapú forgalomirányítás:

Minden router-nek egy táblázatot kell karbantartania,amelyben minden célhoz szerepel a legrövidebb ismert távolság, s annak a vonalnak az azonosítója, amelyiken a célhoz lehet eljutni. A táblázatokat a szomszédoktól származó információk alapján frissítik.

- Elosztott Bellman-Ford forgalomirányítási algoritmusként is nevezik.
- ARPANET eredeti forgalomirányító algoritmusa ez volt. RIP (Routing Information Protocol) néven is ezt használták.

Távolságvektor alapú forgalomirányítás, Elosztott Bellman-Ford algoritmus

Környezet és Működés:

- Minden csomópont csak a közvetlen szomszédjaival kommunikálhat.
- Aszinkron működés.
- Minden állomásnak van saját távolság vektora. Ezt periodikusan elküldi a direkt szomszédoknak.
- A kapott távolság vektorok alapján minden csomópont új táblázatot állít elő.

Végtelenig számolás problémája:

- A „jó hír” gyorsan terjed.
- A „rossz hír” lassan terjed.
- Azaz ciklusok keletkezhetnek.
- Lehetséges megoldás:
  - „split horizon with poisoned reverse”: negatív információt küld vissza arról a szomszédjának, amit tőle „tanult”. (RFC 1058)

Link-state

Algoritmus

1. Szomszédok felkutatása, és hálózati címeik meghatározása

2. Megmérni a késleltetést vagy költséget minden szomszédhoz
3. Egy csomag összeállítása a megismert információkból
4. Csomag elküldése az összes többi router-nek
5. Kiszámítani a legrövidebb utat az összes többi router-hez (Dijkstra algoritmusát használják).

Address Resolution Protocol (ARP)

Feladata:

- Az IP cím megfeleltetése egy fizikai címnek.

Hozzárendelés:

- Adatszóró csomag kiküldése az Ethernetre „Ki-é a 192.60.34.12-es IP-cím?” kérdéssel az alhálózaton, és mindenegyes hoszt ellenőrzi, hogy övé-e a kérdéses IP-cím. Ha egyezik az IP a hoszt saját IP-jével, akkor a saját Ethernet címével válaszol. Erre szolgál az ARP.
- Opcionális javítási lehetőségek:
  - a fizikai cím IP hozzárendelések tárolása (cache használata);
  - Leképezések megváltoztathatósága (időhatály bevezetése);
- Mi történik távoli hálózaton lévő hoszt esetén?
  - A router is válaszoljon az ARP-re a hoszt alhálózatán. (proxy ARP)
  - Alapértelmezett Ethernet-cím használata az összes távoli forgalomhoz

Bitbeszúrás

- Minden keret speciális bitmintával kezdődik és végződik (hasonlóan a bájt beszúráshoz)
  - A kezdő és záró bitsorozat ugyanaz
  - Például: 01111110 a High-level Data Link Protocol (HDLC) esetén
- A Küldő az adatban előforduló minden 11111 részsorozat elé 0 bitet szúr be
  - Ezt nevezzük bit beszúrásnak
- A Fogadó miután az 11111 részsorozattal találkozik a fogadott adatban:
  - 111110 -> eltávolítja a 0-t (mivel ez a beszúrás eredménye volt)
  - 111111 -> ekkor még egy bitet olvas
    - 1111110 -> keret vége
    - 1111111 -> ez hiba, hisz ilyen nem állhat elő a küldő oldalon. Eldobjuk a keretet!
- Hátránya: legrosszabb esetben 20% teljesítmény csökkenés

Rekurzív és iteratív domainnév keresése

- A lekérdezésnek két fajtája van:
  - Rekurzív lekérdezés –> Ha a névszerver végzi el a névfeloldást, és tér vissza a válasszal.
  - Iteratív lekérdezés –> Ha a névszerver adja vissza a választ vagy legalább azt, hogy kitől kapható meg a következő válasz.
- Melyik a jobb?
  - Rekurzív jellemzői

        - Lehetővé teszi a szervernek a kliens terhelés kihelyezését a kezelhetőségért.
        - Lehetővé teszi a szervernek, hogy a kliensek egy csoportja felett végezzen cachelést, a jobb teljesítményért.
    - Iteratív jellemzői
        - Válasz után nem kell semmit tenni a kéréssel a névszervernek.
        - Könnyű magas terhelésű szervert építeni.

Rekurzív DNS lekérdezés:

- A lokális szerver terhet rak a kérdezett névszerverre (pl.root)
- Honnan tudja a kérdezett, hogy kinek továbbítsa a választ?
    - Random ID a DNS lekérdezésben

Iteratív DNS lekérdezés:

- A szerver mindig a következő kérdezendő névszerver adataival tér vissza
    - "I don't know this name, but this other server might"
- Napjainkban iteratív módon működik a DNS!!!

Feszítőfa

Algoritmus

1. Az egyik bride-et megválasztjuk a fa gyökerének
2. Minden bridge megkeresi a legrövidebb utat a gyökérhez
3. Ezen utak unióját véve megkapjuk a feszítőfát

- A fa építése során a bridge-ek egymás között konfigurációs üzeneteket (Configuration Bridge Protocol Data Units [BPDUs]) cserélnek
    - A gyökér elem megválasztásához
    - A legrövidebb utak meghatározásához
    - A gyökérhez legközelebbi szomszéd (next hop) állomás és a hozzá tartozó port azonosításához
    - A feszítőfához tartozó portok kiválasztása
- Kezdetben minden állomás feltételezi magáról, hogy gyökér - Bridge-ek minden irányba szétküldik a BPDU üzeneteiket:
    - | Bridge ID | Gyökér ID | Út költség a gyökérhez |
- A fogadott BPDU üzenet alapján, minden switch választ:
    - Egy új gyökér elemet (legkisebb ismert Gyökér ID alapján)
    - Egy új gyökér portot (melyik interfész megy a gyökér irányába)
    - Egy új kijelölt bridge-et (a következő állomás a gyökérhez vezető úton)

TCP

Lassú indulás - Slow Start

- Cél, hogy gyorsan elérjük a könyök pontot
- Egy kapcsolat kezdetén (vagy újraindításakor)
    - cwnd = 1
    - ssthresh = adv_wnd

  - Minden nyugtázott szegmensre: cwnd++
- Egészen addig amíg
  - El nem érjük az ssthresh értéket
  - Vagy csomagvesztés nem történik
- A Slow Start valójában nem lassú
  - cwnd exponenciálisan nő

Számos TCP változat:

- Tahoe: (az eredeti)
  - Slow start és AIMD
  - Dinamikus RTO, RTT becsléssel
- Reno:
  - fast retransmit (3 dupACKs)
  - fast recovery (cwnd = cwnd/2 vesztés esetén)
- NewReno: javított gyors újraküldés
  - Minden egyes duplikált ACK újraküldést vált ki
  - Probléma: >3 hibás sorrendben fogadott csomag is újraküldést okoz (hibásan!)
- Vegas: késleltetés alapú torlódás elkerülés

TCP jellemzői:

- Kapcsolatorientált:
  - Két résztvevő, ahol egy résztvevőt egy IP-cím és egy port azonosít.
  - A kapcsolat egyértelműen azonosított a résztvevő párral.
  - Nincs se multi-, se broadcast üzenetküldés.
  - A kapcsolatot fel kell építeni és le kell bontani.
  - Egy kapcsolat a lezárásáig aktív.
- Megbízható:
  - Minden csomag megérkezése nyugtázásra kerül.
  - A nem nyugtázott adatcsomagokat újraküldik.
  - A fejléchez és a csomaghoz ellenőrzőösszeg van rendelve.
  - A csomagokat számozza, és a fogadónál sorba rendezésre kerülnek a csomagok a sorszámaik alapján.
  - Duplikátumokat törli.
- Kétirányú Bájtfolyam:
  - Az adatok két egymással ellentétes irányú bájt-sorozatként kerülnek átvitelre.
  - A tartalom nem interpretálódik.
  - Az adatcsomagok időbeli viselkedése megváltozhat: átvitel sebessége növekedhet, csökkenhet, más késés, más sorrendben is megérkezhetnek.
  - Megpróbálja az adatcsomagokat időben egymáshoz közel kiszállítani.
  - Megpróbálja az átviteli közeget hatékonyan használni.

Kapcsolat felépítés:

- Miért van szükség kapcsolat felépítésre?

  - Állapot kialakítása mindkét végponton
  - Legfontosabb állapot: sorszámok/sequence numbers
  - Az elküldött bájtok számának nyilvántartása
  - Véletlenszerű kezdeti érték
- Fontos TCP flag-ek/jelölő bitek (1 bites)
  - SYN – szinkronizációs, kapcsolat felépítéshez
  - ACK – fogadott adat nyugtázása
  - FIN – vége, kapcsolat lezárásához

Kapcsolat felépítés problémája:

- Kapcsolódási zűrzavar
  - Azonos hoszt kapcsolatainak egyértelműsítése
  - Véletlenszerű sorszámmal - biztonság
- Forrás hamisítás
  - Kevin Mitnick
  - Jó random szám generátor kell hozzá!
- Kapcsolat állapotának kezelése
  - Minden SYN állapotot foglal a szerveren
  - SYN flood = denial of service (DoS) támadás
  - Megoldás: SYN cookies

Kapcsolat lezárása:

- Mindkét oldal kezdeményezheti a kapcsolat bontását
- A másik oldal még folytathatja a küldést
  - Félig nyitott kapcsolat
  - shutdown()
- Az utolsó FIN nyugtázása
  - Sorszám + 1