Tehtävä 2: Tietoliikenneprotokolla

Johdanto

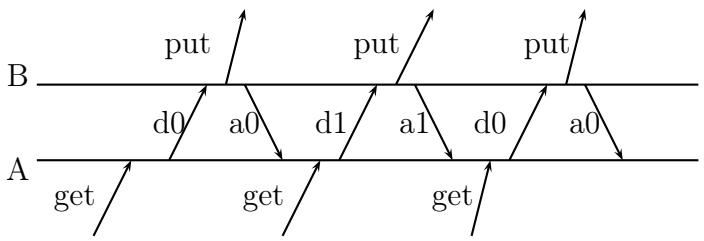
Tarkastellaan tilannetta, jossa tietokone A lähettää datapaketteja tietokoneelle B tiedonsiirtovirheille alttiin kanavan kautta. Datapaketit ovat biteistä eli nollista ja ykkösistä koostuvia jonoja. Tiedonsiirtovirheitä on kahdenlaisia: joko paketti katoaa kokonaan tai paketin bittejä on muuttunut (nolla ykköseksi tai ykkönen nollaksi), jolloin puhutaan *paketin vääristymisestä*. Normaali tilanne on sellainen, että tiedonsiirtovirheitä tapahtuu harvakseltaan.

Tiedonsiirtoon liittyy kaksi nopeutta. Ensinnäkin tietokone voi syöttää kanavaan korkeintaan tietyllä maksimaalisella nopeudella. Tämä nopeus, *kanavan kapasiteetti* (yksikkönä bittejä sekunnissa), riippuu kanavan ominaisuuksista (ja osittain myös tietokoneen rakenteesta, joskin yleensä kanavan ominaisuus on ratkaiseva). Toinen nopeus liittyy bittien etenemiseen kanavassa. Tämä nopeus on yleensä sama kuin valon nopeus väliaineessa (km/s). Jos paketteja siirretään hyvin pitkiä matkoja (tuhansia kilometrejä), niin matkan aiheuttama viive alkaa olla sellainen, että se joissakin tilanteissa täytyy ottaa huomioon. Näiden nopeuksien avulla voidaan laskea, kuinka kauan kestää paketin lähettäminen ja vastaanottaminen.

Vuorottelevan bitin protokolla

Tiedonsiirtoon tietokoneet käyttävät seuraavaa algoritmia eli protokollaa (ns. *vuorottelevan bitin protokolla*). Protokolla toimii lähettäjän puolella siten, että se ottaa vastaan datapaketteja sovellukselta ja lähettää niitä sen jälkeen kanavaan. Vastaanottajan puolella protokolla ottaa vastaan paketteja kanavasta ja luovuttaa ne sovellukselle. Käytännössä lähettävä sovellus sijoittaa paketteja varatulle muistialueelle eli *puskuriin*, josta protokolla käy niitä noutamassa. Samoin vastaanottajan puolella protokolla sijoittaa vastaanotetut paketit puskuriin, josta sovellus käy ne noutamassa.

Oletetaan seuraavassa, että lähettävä kone on A ja vastaanottava kone B. Kun A:n lähettämä paketti saapuu virheettömänä B:lle, B kuittaa paketin vastaanotetuksi *lähettämällä kuittaussanoman* A:lle. Vasta kun kuittaus on saapunut A:lle, A voi ottaa puskurista uuden paketin ja lähettää sen. Jotta kuittaussanoma kohdistuisi aina oikeaan pakettiin, paketit ja kuittaukset numeroidaan. Osoittautuu, että vain numeroita 0 ja 1 tarvitsee käyttää. Seuraava kaavio näyttää protokollan toiminnan, kun tiedonsiirtovirheitä ei tapahdu.



Eli A ottaa ensin get-sanomalla datapaketin puskurista ja lähettää datapaketin d varustettuna numerolla 0 ja jää odottamaan kuittausta. B saa datapaketin virheettömänä, luovuttaa sen put-sanomalla sovellukselle eli puskuriin ja lähettää kuittauksen a varustettuna myös numerolla 0 jääden odottamaan seuraavaa pakettia. A vastaanottaa virheettömän kuittauksen oikealla numerolla 0, ottaa uuden paketin puskurista ja lähettää tämän jälkeen uuden datapaketin varustettuna nyt numerolla 1. B vastaanottaa tämän paketin, tunnistaa sen uudeksi paketiksi numeron perusteella, luovuttaa paketin sovellukselle ja lähettää paketin kuittauksen uudella numerolla 1. Tämäkin kuittaus tulee virheettömänä perille. A tunnistaa sen numeron perusteella edellisen paketin kuittaukseksi, ottaa taas uuden paketin puskurista ja lähettää tämän uuden datapaketin, johon tulee taas numero 0. B vastaanottaa, luovuttaa eteenpäin ja kuittaa.

Jatkossa tällaista kaaviota, joka esittää jotain tilannetta tiedonsiirrossa, kutsutaan *skenaarioksi*. Tämän protokollan skenaarioissa siis aika kulkee vasemmalta oikealle. Vaakasuorien viivojen väliin jäävä alue kuvaa tilanetta, kun sanomat tai kuittaukset ovat kanavassa. Put- ja get-sanomat hakevat tai luovuttavat sanomia puskuriin. Skenaarioon ei ole piirretty puskureita erikseen.

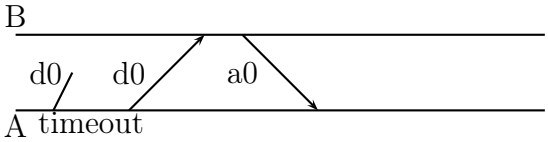
Protokollan toiminta, kun kanava kadottaa tai vääristää sanomia

Edellisessä skenaariossa tiedonsiirtokanava toimi virheettömästi. Protokollan täytyy kuitenkin varautua myös virheisiin. Tavallisia virheitä ovat tilanteet, joissa paketti katoaa tai vääristyy, tai kuittaus katoaa tai vääristyy. Oletetaan, että tiedonsiirtokanava voi myös monistaa paketteja. Tämä tarkoittaa, että kun lähetetään paketti, kanava toimittaakin perille kaksi tai useampia kopioita samasta paketista. Eri kopioiden välillä voi olla mielivaltaisen pitkiä taukoja. Tiedonsiirtokanavasta oletetaan lisäksi, että kanavassa voi olla useita paketteja samaan aikaan, mutta pakettien järjestys kanavassa ei voi muuttua. Toisin sanoen jos kanavaan lähetetään paketit m_1 ja m_2, niin paketit tulevat perille myös samassa järjestyksessä. Tämä koskee myös monistumista: Jos kanava

monistaa sanoman m sanomiksi m_1 ja m_2, niin ne saapuvat perille samassa järjestyksessä. Mikään toinen paketti ei voi saapua perille m_1:n ja m_2:n välissä.

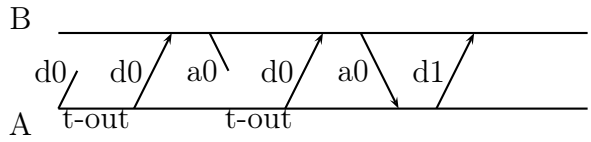
Jotta protokolla toipuisi tiedonsiirtovirheistä, otetaan käyttöön *ajastin*. Ajastin on lähettäjän A puolella. Mikäli A ei saa kuittausta kohtuullisessa ajassa, ajastin laukeaa eli protokolla saa *timeout*-sanoman (tai t-out -sanoman) ja A lähettää edellisen datapaketin uudestaan. Vääristyneet datasanomat ja kuittaukset tulkitaan kadonneiksi paketeiksi, toisin sanoen vääristyneisiin paketteihin ei reagoida. Protokollan täytyy toimia siten, ettei se luovuta eteenpäin jo aikaisemmin tullutta ja eteenpäin luovutettua pakettia. (Tällainen protokolla on osa laajempaa ohjelmistokokonaisuutta. Muut ohjelmiston osat ottavat datapaketit tarkempaan käsittelyyn ja protokollamme tehtävänä on vain ottaa vastaan lähetettäviä paketteja muilta ohjelmistomoduuleilta, välittää paketit tiedonsiirtokanavan yli ja luovuttaa vastaanottopäässä paketit edelleen muille moduuleille.)

Katsotaan seuraavaksi skenaarion avulla, miten protokolla toipuu, jos datapaketti katoaa. Tässä ja seuraavissa skenaarioissa jätetään get- ja put-sanomat kirjoittamatta selvyyden vuoksi. Muistettakoon kuitenkin, että aina kun uusi paketti lähetetään, se on ensin otettava lähetyspuskurista get-operaatiolla. Samoin kun uusi paketti otetaan vastaan, se luovutetaan ensin put-sanomalla sovellukselle ja vasta sen jälkeen lähetetään kuittaus. Jos tulee kaksi kertaa peräkkäin samalla numerolla varustettu sanoma, niin toinen on uusintalähetys, eikä sitä luovuteta enää sovellukselle toistamiseen.



Skenaariossa ensimmäinen datapaketti $d0$ katoaa. Vastaanottaja B jatkaa odottamistaan, mutta jossain vaiheessa A:n ajastin laukeaa, koska kuittausta ei kuulu. A lähettää saman paketin $d0$ uudestaan ja nyt se menee perille B:lle. B kuittaa, kuittaus menee perille vääristymättä ja nyt ollaan taas normaalissa tilanteessa ja A voi jatkaa lähettämällä uuden datapaketin numerolla 1. Tilanne, jossa kuittaus katoaa, on täysin vastaava. A odottaa kuittausta, jota ei tule, ja sen johdosta A:n ajastin laukeaa. A lähettää edellisen datapaketin uudestaan. Jos se menee nyt vääristymättä perille B:lle, B huomaa numerosta, että se on vanha paketti, eikä luovuta sitä enää eteenpäin. Sen sijaan B lähettää uudestaan kuittauksen. Jos se menee vääristymättä perille, tilanne jatkuu taas normaalisti seuraavan datapaketin lähetyksellä.

Katsotaan vielä skenaario, jossa datapaketti katoaa ja heti sen jälkeen kuittauskin katoaa. Protokolla toipuu tästäkin tilanteesta ajastimen avulla:



Skenaariossa ensimmäinen datapaketti $d0$ katoaa, joten kuittausta ei kuulu ja A:n ajastin laukeaa. A lähettää datapaketin $d0$ uudestaan ja tällä kertaa se menee perille. Kuitenkin pakettiin liittyvä kuittaus $a0$ katoaa. Täten A:n ajastin laukeaa taas ja datapaketti $d0$ lähetetään uudestaan. Tämäkin datapaketti menee virheettömänä perille. B huomaa saman paketin saapuneen uudelleen, ei luovuta sitä eteenpäin, mutta lähettää kuittauksen. Nyt kuittaus $a0$ menee virheettömänä perille ja protokolla jatkuu normaalisti seuraavan paketin lähetyksellä.

Kysymykset

Kysymys 2.1.
Näytä skenaarion avulla, miten protokolla käsittelee tilanteen, jossa kuittaus vääristyy. Piirrä tähän ja seuraaviin skenaarioihin myös get- ja put-sanomat. (maksimipistemäärä 2)

Kysymys 2.2.
Näytä skenaarion ja siihen liittyvän sanallisen selostuksen avulla, ettei protokolla toimi oikein, jos sanomia ei numeroida. (maksimipistemäärä 3)

Kysymys 2.3.
Oletetaan, että ajastin laukeaa, vaikka kuittaus onkin tulossa virheettömänä A:lle. Näytä skenaarion avulla, että protokolla toipuu tästäkin tilanteesta. (maksimipistemäärä 4)

Kysymys 2.4.
Osoita skenaarion avulla, ettei protokolla toimi oikein, jos kanava voi kadottaa ja monistaa sanomia sekä lisäksi vaihtaa sanomien järjestystä kanavassa. (maksimipistemäärä 4)

Käännä!

Kysymys 2.5.
Oletetaan, että protokollaa käytetään siirtämään 10 megatavun tiedostoa koneelta A koneelle B. Koneiden välimatka on 4000 km ja signaalin nopeus kanavassa (yksinkertaisuuden vuoksi) 200 000 km/s. Kanavaan voi lähettää nopeudella 10 megabittiä sekunnissa. Kuinka kauan tiedoston lähettäminen kestää, kun käytetään vuorottelevan bitin protokollaa ja tiedosto pilkotaan 1000 tavun kokoisiksi paketeiksi, jotka lähetetään erikseen. Kuittausten koko on 1000 bittiä. Kanava toimii virheettömästi.

Tavu on 8 bittiä, kilotavu on (yksinkertaisuuden vuoksi) 1000 tavua, megatavu on 1000 kilotavua. (maksimipistemäärä 5)

Kysymys 2.6.
Kuten edellinen kysymys, mutta nyt kanava kadottaa 10% sanomista, joista puolet on datasanomia, puolet kuittauksia. Ajastin laukeaa 5 sekunnin kuluttua, jos kuittausta ei ole saapunut. (maksimipistemäärä 5)

Kysymys 2.7.
Selitä, mitä tekijöitä täytyy ottaa huomioon ajastimen ajan asetuksessa, kun tavoitteena on tehokas ajastin, joka ei laukea liian aikaisin, mutta joka ei myöskään jätä odottamaan turhan kauan. (maksimipistemäärä 2)