


LUENTO 20

# Hajautettu prosessien hallinta

## Stallings, Luku 15

1



## Sisältöä luento 20

- Prosessien siirto (Process Migration)
- Globaali tila (hajautetussa järjestelmässä)
  - Tila ja snapshot
  - Ota kantaa toipumiseen (ja peruutuksiin)
- Hajautettu poissulkeminen
  - Lamportin kellot – tapahtuminen järjestys
- Hajautettu lukkiuma (ja sen havaitseminen)

Tarkemmin hajautettujen järjestelmien kurssilla

2

## Prosessin siirto

- Prosessi vaihtaa konetta kesken suorituksen:
  - Suoritus keskeytetään siirron ajaksi
  - Siirretään uuteen paikkaan riittävästi tietoa nykyisestä tilasta.
  - Suoritus jatkuu uudessa paikassa
- Mikä on riittävästi tietoa?
  - Prosessin tila (prosessin kuvaaja, PCB)
  - Osoiteavaruus (muistin sisältö)
  - Avoimet tiedostot
  - Muut tiedot
- Siirretäänkö kaikki vai vain tarvittava osa?

3

## Syitä siirtoon

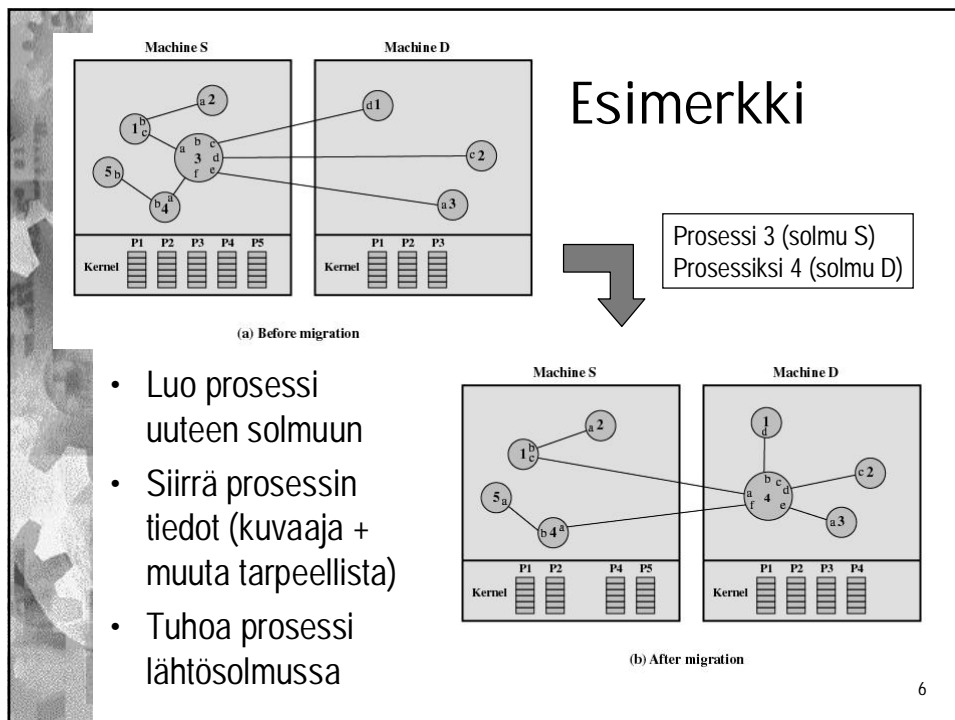
- Kuorman tasaus (Load sharing)
- Toiminnan tehostaminen
  - Keskenään kommunikoivat prosessit samaan paikkaan
  - Siirretään prosessi sinne missä käsiteltävä data on
- Saatavuus
  - Prosessi on siirrettävä, jotta laskenta voi jatkua vaikka solmu ei olekaan (enää) käytettävissä
- Erikoistarpeet (poikkeuksellisia resursseja)
  - Prosessi siirretään sinne, missä on sen tarvitsemia erikoislaitteistoja tai -ohjelmistoja

4

## Kuka/mikä päättää siirrosta?

- Käyttöjärjestelmä
  - Esimerkiksi kuorman tasaus
- Prosessi itse
  - Tarvitaan erikoisresursseja tai tiettyä dataa

5



## Siirtopolitiikkoja

- Eager (all): Siirrä kaikki (muistialueet+muut)
  - Mitä ei jätetä lähtöpaikkaan
  - Kallis, jos kaikkea ei tarvita
- Precopy: Prosessia vielä suoritetaan, kun muistialueita jo kopioidaan
  - Muutetut sivut täytyy kopioida toistamiseen lopuksi
  - Lyhentää prosessin 'jäädetyksen' kestoja, eli aikaa, jolloin prosessia ei voida suorittaa

7

## Siirtopolitiikkoja

- Eager (dirty): Siirrä vain se osa muistiavaruutta, joka on keskusmuistissa ja jota on muutettu
  - Loput sivut haettava sivunpuutoksen kautta
  - Jos ei jaettua levyä, niin lähdekone toimittaa puuttuvia sivuja aina pyynnöstä
- Copy-on-reference: Siirrä sivu vasta viitattaessa
  - Ei juurikaan viivettä prosessin siirtovaiheessa
- Flushing: Kopioi prosessin muuttuneet sivut levyille
  - Lähdekoneen ei tarvitse säilyttää sivuja keskusmuistissaan

8

## Siirron aloitus: Neuvottelu

- Esimerkkinä Charlotte (kts. IEEE Computer, syysk. 1989)
- Siirtopolitiikasta vastaa ja neuvottelun aloittaa lähdekoneen Starter-palvelu
- Starter-palvelun tehtäviin kuuluu myös muistinhallinta ja prosessin hyväksyminen järjestelmään (long-term scheduling)
- Siirto toteutuu vasta kun sekä lähde- että kohdekoneen Starter-palvelut hyväksyvät siirron
- Vastaanottajalla on oikeus kieltäytyä

9

## Neuvottelu siirrosta: Charlotte

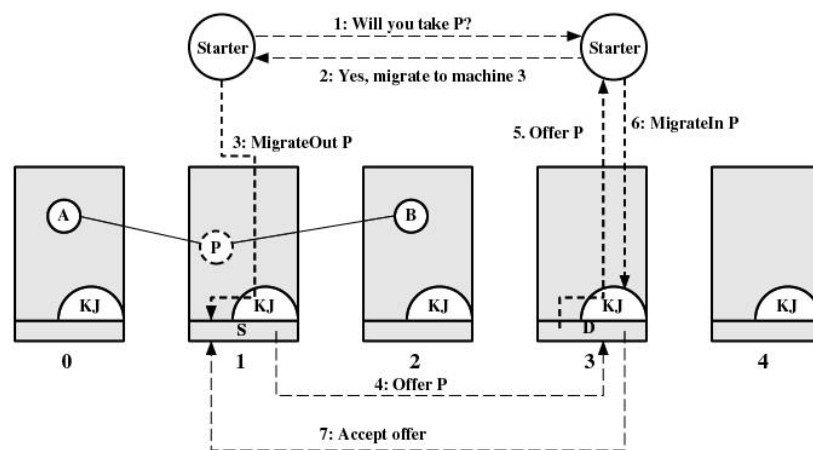


Figure 15.2 Negotiation of Process Migration

## Häätö (Eviction)

- Jos siirtoja tehdään kuorman tasaamiseksi, niin
- Joutilaaseen työasemaan (idle workstation) voidaan siirtää prosesseja
- Kun työasemalle tulee paikallista kuormaa, voi olla tarpeen häätää siirrettyjä prosesseja vaikkapa takaisin lähtöpisteeseen (kotiin) tai jonnekin muualle
  - Tarvitaan esim. vasteaikojen pitämiseksi hyväksyttävänä

11

## Globaali tila

12

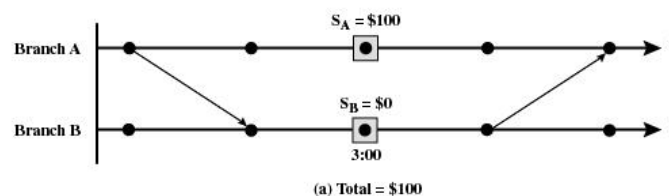
## Tilatieto

- Keskitetyssä järjestelmässä tilatieto on saatavilla
- Hajautetussa järjestelmässä ei ole mahdollista täsmällisesti tietää koko järjestelmän tilaa
  - Käyttöjärjestelmä tai yksittäinen prosessi ei voi tietää kaikkien prosessien sen hetkistä tilaa
  - Prosessi voi tietää vain muiden paikallisten prosessien tilan
  - Kauempana olevien tila voidaan päätellä vain saapuvista viesteistä, jotka kertovat tilasta jonkin aika sitten

13

## Esimerkki

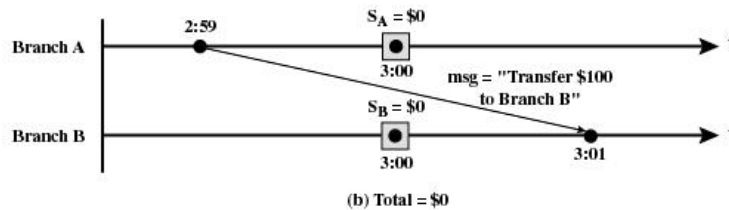
- Pankkitili on jaettu kahden toimipisteen kesken
- Tilin kokonaissaldo on näiden osatilien summa
- Tilin saldo päätellään täsmälleen ajanhetkellä 3:00 PM
- Osasaldojen kyselyt perustuvat sanomiin



14

## Esimerkki: liikkuvat viestit

- Jos juuri saldokyselyn aikaan, rahaa siirretään tililtä toiselle, niin tulos voi olla virheellinen



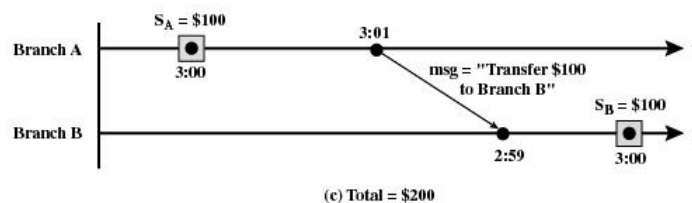
Korjaus:

- Kaikki siirrettävänä olevat viestit täytyy myös tarkistaa
- Kokonaissaldoon vaikuttavat osatilien lisäksi, myös viesteissä olevat summat

15

## Esimerkki: kellovirhe

- Jos kellot eivät ole synkronissa ja kysely tehdään tasan klo 3:00, niin voi käydä seuraavasti
- Siirretään rahaa paikasta A klo 3:01
- Tieto saapuu paikkaan B klo 2:59 (siis etuajassa!)
- Siirretty summa lasketaan kokonaissaldoon kahdesti



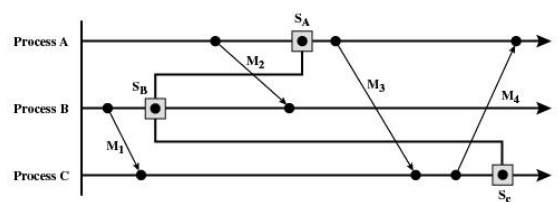
16

## Termejä

- Kanava (Channel)
  - Kahden kommunikoivan prosessin välillä viestien vaihtoon
- Tila (State)
  - Prosessin kanaviaan pitkin lähettämien ja vastaanottamien viestin jono
- Globaali tila (Global state)
  - Prosessien yhdistetty tila
- Snapshot - tilannevedos
  - Prosessin tilan 'tietyllä hetkellä'
- Distributed Snapshot
  - Kooste prosessien tiloista 'tietyllä hetkellä'

17

## Inconsistent Global State

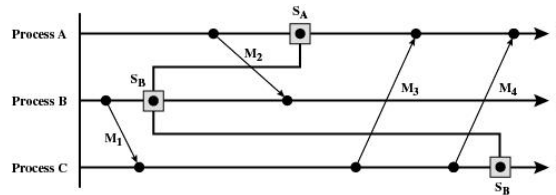


(a) Inconsistent Global State

- Globaalissa tilassa on ristiriitoja, se ei ole eheä
- Tässä tapauksessa viesti M3 ylittää globaalialta tilaa kuvaavan rajan väärään suuntaan. (Se näyttäisi saapuneen tulevaisuudesta)

18

## Consistent Global State



(b) Consistent Global State

19

## Distributed Snapshot Algorithm

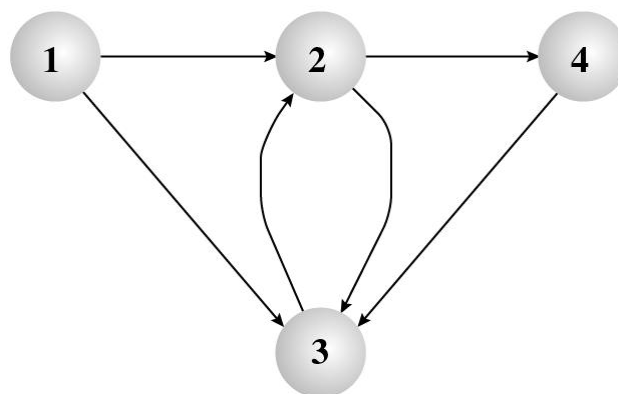


Figure 15.5 Process and Channel Graph

20

## Algoritmi

- Aloittaja:
  - Tallenna oma tila
  - Lähetä marker-viesti muille
- Marker-viestin vastaanottaja:
  - Ensimmäinen kerta (atominen toimenpidesarja)
    - Tallenna oma tila ja tilaan kirjataan tämän viestin saapumiskanava tyhjäksi
    - Lähetä oma tila kaikille muille
  - Muut viestit (muista kanavista, yksi per kanava)
    - Kanavan tilaksi kootaan kaikki viestit, jotka ovat tulleet oman lähetyksen ja tämän viestin välissä
- Algoritmi päättyy äärellisessä ajassa, kunhan viestejä ei voi kadota ja viestien kulkuaika on äärellinen
- Globaali tila saadaan näistä prosessi- ja kanavakohtaisista tiloista
  - Esimerkiksi aloittaja voi koota ne pyytämällä tai prosessit lähettävät ne automaattisesti

21

## Distributed Snapshot Algorithm

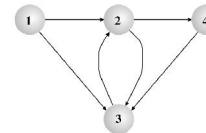


Figure 15.5 Process and Channel Graph

<p><b>Process 1</b></p> <p>Outgoing channels</p> <p>2 sent 1, 2, 3, 4, 5, 6</p> <p>3 sent 1, 2, 3, 4, 5, 6</p> <p>Incoming channels</p>	<p><b>Process 3</b></p> <p>Outgoing channels</p> <p>2 sent 1, 2, 3, 4, 5, 6, 7, 8</p> <p>Incoming channels</p> <p>1 received 1, 2, 3 stored 4, 5, 6</p> <p>2 received 1, 2, 3 stored 4</p> <p>4 received 1, 2, 3</p>
<p><b>Process 2</b></p> <p>Outgoing channels</p> <p>3 sent 1, 2, 3, 4</p> <p>4 sent 1, 2, 3, 4</p> <p>Incoming channels</p> <p>1 received 1, 2, 3, 4 stored 5, 6</p> <p>3 received 1, 2, 3, 4, 5, 6, 7, 8</p>	<p><b>Process 4</b></p> <p>Outgoing channels</p> <p>3 sent 1, 2, 3</p> <p>Incoming channels</p> <p>2 received 1, 2 stored 3, 4</p>

22

## Poissulkeminen hajautetussa järjestelmässä

### Perusominaisuuksia

- Poissulkeminen on välttämätöntä ja onnistuttava: vain yksi prosessi kerrallaan saa olla kriittisellä alueella
- Prosessi, joka suoritus päättyy (ei-kriittisellä alueella), ei saa häiritä muiden toimintaa
- Kriittiselle alueelle pääsyä ei saa joutua odottamaan äärettömän kauan: ei siis sallita nälkiintymistä tai lukkoja

23

## Poissulkeminen hajautetussa järjestelmässä

Lisää: RiO ja  
Hajautetut järjestelmät

- Oletukset:
  - Ei jaettua muistia
  - Ei jaettua (synkronoitua) kelloa
  - Kaikki tieto kulkee vain viesteissä
  - Varattavaa asiaa kutsutaan kriittiseksi alueeksi
    - Vain yksi kerrallaan, varauksen kesto on äärellinen
- Ongelma:
  - Poissulkeminen ilman nälkiintymistä tai lukkiutumaa

24

## Malli

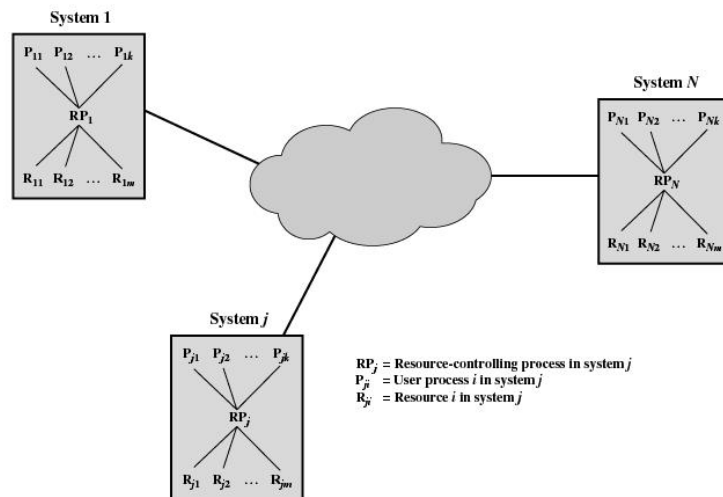


Figure 15.7 Model for Mutual Exclusion Problem in Distributed Process Management

25

## Keskitetty algoritmi

- Nimitetään yksi solmu vastuulliseksi kontrollisolmuksi
- Se myöntää luvat ja pitää kirjaa varauksista
- Vain sillä on tieto varauksista, mutta
- Jos se kaatuu, niin varaukset katoavat eikä kukaan enää tiedä ketkä voivat käyttää resursseja ja ketkä eivät
- Ongelmia:
  - Single point of failure
  - Saattaa muodostua pullonkaulaksi

26

## Hajautettu algoritmi

- Kaikilla solmuilla suurin piirtein saman verran tietoa globaalista tilasta
- Kullakin on vain osittainen kuva tilasta ja se joutuu tekemään päätökset tämän puutteellisen tiedon varassa
- Kaikilla solmuilla on samanlainen päätösvalta ja ne tekevät suurin piirtein saman verran töitä päätöksen eteen
- Yksittäisen solmun kaatuminen ei yleensä kadota liikaa tietoa, kokonaisuus voi edelleen toimia
- Ongelma:
  - Ei synkronoitua kelloa
  - sanomien kulkuajat vaihtelevat
  - niiden saapumisjärjestys eri solmuihin voi vaihdella

27

## Tapahtumien järjestäminen

- Sanomien järjestyksellä on merkitystä prosessien käsitykselle globaalista tilasta
  - Tilan määritelmään oli prosessien lähettämät ja vastaanottamat sanomat
- Jotta hajautettujen prosessien tilakäsitykset vastaisivat toisiaan, on sanomat saatava jollain tavalla järjestykseen.
- Käsitellään tapahtumina vain sanomien lähettämiset
- Lamport: Time, clocks and the ordering of events, CACM, July 1978 - esittää yhden mahdollisuuden

28

## Tapahtumien järjestäminen

- Tapahtumat (sanomien lähettämiset) on järjestettävä, jotta vältetään lukkiumat ja taataan poissulkeminen
  - Varaamiset ja vapauttamiset tehdään sanomien avulla
- Ongelmia: kellojen vaihtelu, kommunikointiviipeet
- Voidaan käyttää loogisia aikaleimoja (Time-stamping)
  - Näiden avulla saadaan tapahtumat järjestettyä
  - Eivät siis kellonaikoja vaan sanomien numeroiteja

29

## Aikaleimat

- Jokaisessa solmussa on oma laskuri (virtuaalikello), josta saadaan aikaleimat
  - Näillä saadaan osittaisjärjestys
- Lisäksi solmut on numeroitu
  - Näistä saadulla lisätiedolla saadaan täydellinen järjestys
- Viestien lähetys: liitä viestiin oma tunniste ja oma aikaleima (kasvata aikaleimaa aina ennen lähetystä)
- Kaikki viestit lähetetään kaikille
- Viestien vastaanotto: päivitä oma aikaleima: Uusi arvo on  $1 + \max(\text{oma aikaleima}, \text{saapunut aikaleima})$

30



## Yhden prosessin tilasiirtymät

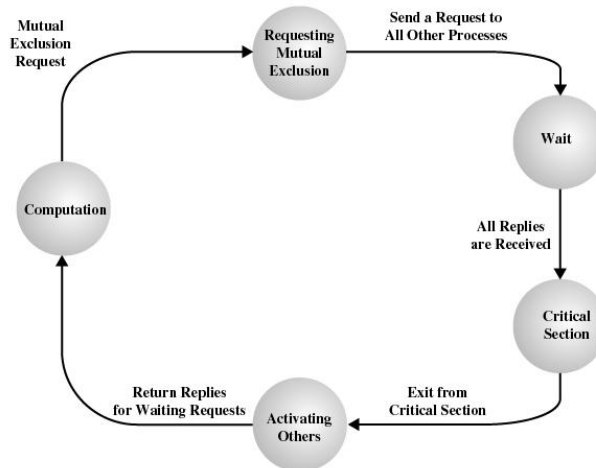


Figure 15.10 State Diagram for Algorithm in [RICA81]

33

## Poissulkeminen käyttäen vuoromerkkiä (Token)

- Jos yksi ja sama kriittinen alue kaikilla (yksi resurssi)
- Siirrä vuoromerkkiä prosessilta toiselle
- Vuoromerkin haltijalla on oikeus suorittaa kriittinen alue, muut joutuvat odottamaan vuoroaan
- Kun prosessi poistuu kriittiseltä alueelta, se siirtää vuoromerkin eteenpäin jollekin toiselle prosessille
- Ongelma: vuoromerkin katoaminen

34

## Lukkiumat (Deadlocks)

35

## Lukkiumat

- Hajautetuissa järjestelmissä lukkiuman syynä voi olla resurssit tai viestit
- Resurssien osalta lukkiuma voi syntyä vain, jos
  - Poissulkeminen (Mutual exclusion)
  - Seuraavaa resurssia voi odottaa ja pitää lukkoa aiemmasta edelleen (Hold and wait)
  - Ei irroittamista (No preemption)
  - Odotuksista muodostuu rengas (Circular wait)
- Lukkiumia voidaan
  - Estää – prevention
  - Välttää – avoidance
  - Havaita – detection

36

## Phantom Deadlock

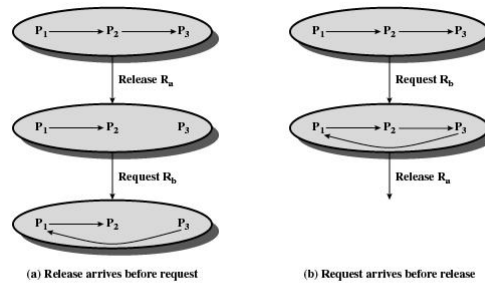


Figure 15.12 Phantom Deadlock

Hajautetuissa järjestelmissä on lukkiamaa etsittäessä syytä huomioida myös jo lähetetyt (uudet) viestit. b) näyttäisi lukkiumalta, mutta ei sitä ole.

37

## Lukkiuman estäminen (Deadlock Prevention)

- Ketjuuntunut odotus voidaan estää, esimerkiksi sallimalla resurssin varaus vain tietyssä järjestyksessä.
- Hold-and-wait voidaan estää vaatimalla kaikkien resurssien varaamista kerralla. Prosessi odottaa kunnes se saa ne kaikki.

Ongelma:

- Nämä ratkaisut toimivat vain kun tiedetään resurssitarpeet etukäteen (viimeistään suorituksen alussa)
- Tietokannoissa käytetään yleensä wait-die tai wound-wait -menetelmiä, joissa ei tarvita ennakkotietoa

38

## Lukkiuman välttäminen (Deadlock Avoidance)

- Jokaisen solmun pitäisi tietää järjestelmän globaali tila koko ajan
- Globaalin tilan turvallisuuden tarkistus pitää tehdä poissulkevasti, jotta tila varmasti on edelleen turvallinen
- Tilan turvallisuuden tarkistus on raskasta, kun järjestelmässä on paljon prosesseja ja resursseja

⇒ Hajautettu lukkiuman välttäminen on epäkäytännöllistä

39

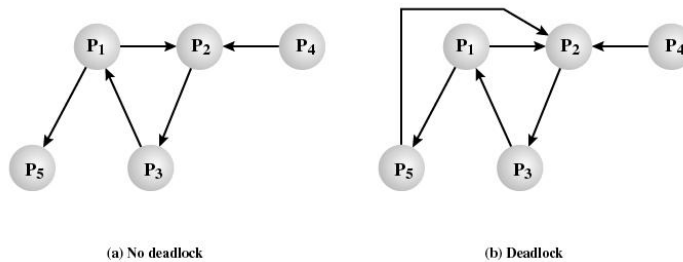
## Hajautettu lukkiuman havaitseminen (Distributed Deadlock Detection)

- Jokainen solmu tuntee vain omat resurssinsa
  - Lukkiuma voi liittyä hajautettuja resursseja
- Keskitetty – yksi solmu vastuussa lukkiuman havaitsemisesta
- Hierarkkinen – havaintovastuu sillä (alimmalla tasolla), jonka alipuussa lukkiutuneet solmut ovat
- Hajautettu – kaikki solmut osallistuvat lukkiuman havaitsemiseen

40

## Lukkiuma sanomanvälityksessä

- Mutual Waiting (odotetaan viestiä toisilta, jotka odottavat)
  - Kaikki lukkiutuneet solmut odottavat viestiä vain muilta samassa odotustilassa olevilta solmuilta ja yhtään sanomaa ei ole liikkeellä.



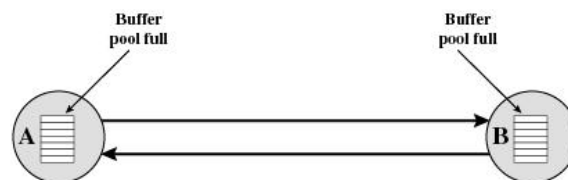
P1 odottaa joko P2 tai P5

Figure 15.16 Deadlock in Message Communication

41

## Lukkiuma sanomanvälityksessä

- Viestipuskurit täynnä
  - Havaittu mm. pakettikytkentäisissä verkoissa
  - Kuvan esimerkki: A:n puskuri on täynnä B:lle meneviä viestejä, joten sinne ei voi vastaanottaa uusi viestejä. B:n tilanne on päinvastainen.

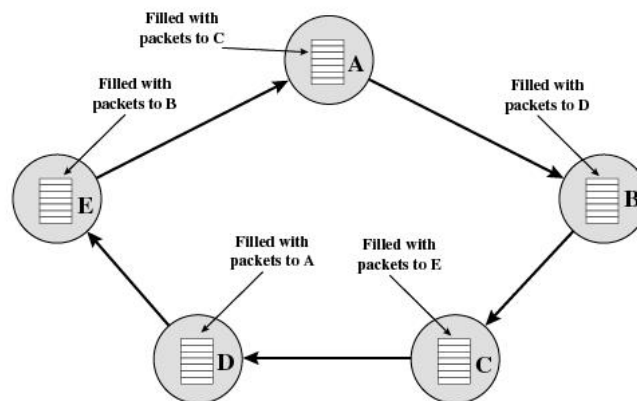


42

## Lukkiuma sanomanvälityksessä

- Viestipuskurit täynnä
  - Tilanne voi olla myös ketju ja monimutkainen.
  - Kuvan esimerkissä kunkin solmun puskurit ovat täynnä parin solmun päähän meneviä viestejä.

43



(b) Indirect store-and-forward deadlock

44