T-79.179 Rinnakkaiset ja hajautetut digitaaliset järjestelmät

Prosessialgebra

Teemu Tynjälä

19. maaliskuuta 2002

Petri-verkot vastaan prosessialgebra

- Petri-verkot esittävät rinnakkaisia ja hajautettuja järjestelmiä graafisesti.
- Valitettavasti kovin suuria järjestelmiä ei voi kuvailla graafisesti. Nykyiset raskaan sarjan Petri-verkkotyökalut käyttävätkin tekstimuotoista esitystapaa järjestelmille.
- Prosessialgebra on jo alkujaan tekstipohjainen tapa kuvata rinnakkaisia ja hajautettuja järjestelmiä.

Prosessialgebra syntyi Isossa Britanniassa 1980-luvulla, ja sen isät olivat Hoare ja Milner. Tällä luennolla käsittelen CCS-prosessialgebraa *(Calculus of Communicating Systems)* , jonka Milner julkaisi vuonna 1980.

Prosessialgebrojen pääpiirteet (1/2)

- Prosessialgebran keskeinen käsite on *agentti*. Agentilla on in-portti, josta se lukee ympäristöstään tulevia viestejä, ja vastaavasti out-portti, johon se tulostaa viestejä.

- Varsin usein agentti toimii seuraavasti:

  1. Agentti lukee in-portistaan tietyn määrän viestejä
  2. Se suorittaa sisäistä laskentaa
  3. Lopuksi agentti saattaa tulostaa jotain ja jää odottamaan uusia viestejä

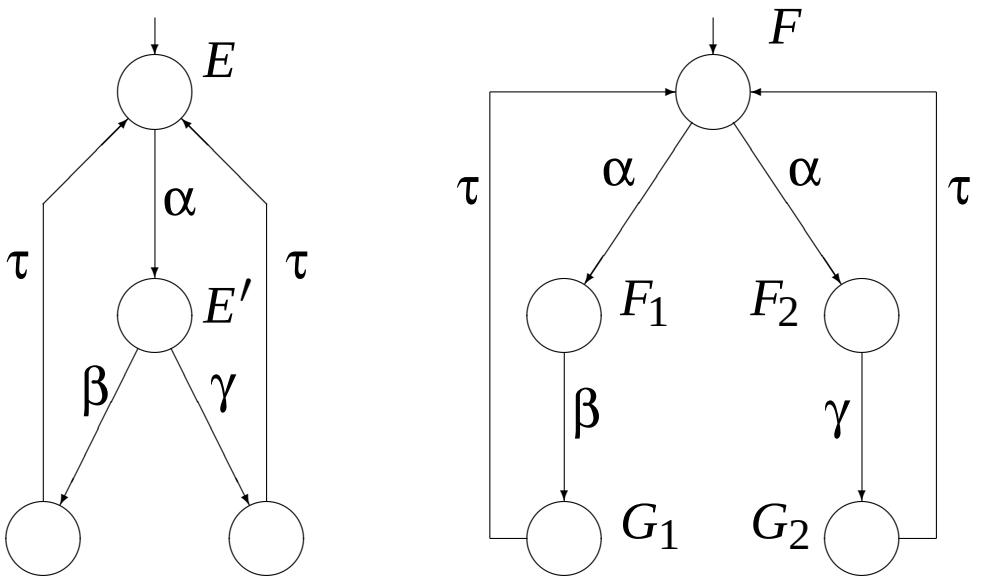
- Yleensä prosessialgebrassa tutkitaan järjestelmiä, jotka koostuvat useasta keskenään kommunikoivasta agentista.

Prosessialgebrojen pääpiirteet (2/2)

- Prosessialgebrojen suurin vahvuus verrattuna Petri-verkkoihin on niiden rakenteellisuus. Prosessialgebran operaattoreita soveltamalla voi muodostaa uusia agentteja esimerkiksi kytkemällä yksinkertaisempia agentteja rinnan.

- Prosessialgebra antaa myös mahdollisuuden todeta, ovatko kaksi agenttia samanlaiset ulkoisen tarkkailijan kannalta.

- Prosessialgebra mahdollistaa agenttien kuvaamisen sekä tekstinä että kaavioina. Seuraavalla kalvolla on esitetty kaksi agenttia molemmilla tavoilla.



Vasemmanpuoleinen agentti voidaan kuvata lausekkeella $E ::= \alpha.(\beta.\tau + \gamma.\tau)$ ja oikeanpuoleinen lausekkeella $F ::= \alpha.\beta.\tau + \alpha.\gamma.\tau$. (Vertaus postimerkkiautomaattiin.)

Kuinka prosessialgebra (CCS) toimii? (1/2)

- Jokaisella agentilla on nimi. Nimet ovat yleisesti isoja kirjaimia, $P, Q, R, A, B, C, \ldots$

- Prosessialgebran täytyy myös määritellä *näkyvien tapahtumien joukko*. Näkyvät tapahtumat ovat sellaisia, joita agentit suorittavat, ja jotka ulkoiset tarkkailijat pystyvät huomaamaan. Yleensä on $Act = \{\alpha, \overline{\alpha}, \beta, \overline{\beta}, \gamma, \overline{\gamma}, \ldots\}$.

- Jokaisella tapahtumalla on *komplementtitapahtuma*. Jos α tarkoittaa viestin α vastaanottamista in-portista, niin sen komplementtitapahtuma $\overline{\alpha}$ merkitsee samaisen viestin kirjoittamista out-porttiin. Pätee myös $\overline{\overline{\alpha}} = \alpha$.

- On myös *näkymätön tapahtuma* τ, jota ulkoinen havainnoitsija ei huomaa.

Kuinka prosessialgebra (CCS) toimii? (2/2)

- Prosessialgebrassa keskeisin käsite on se, miten agentit käyttäytyvät tietyn tapahtuman jälkeen. Jos kirjoitetaan $E \xrightarrow{\alpha} E'$, agentti E käyttäytyy tapahtuman α jälkeen kuten agentti E'.

- Agentti pystyy siirtymään myös näkymättömän tapahtuman avulla. Tällöin kirjoitetaan $G \xrightarrow{\tau} G'$. Nyt ulkopuolinen havainnoitsija ei tiedosta G:n siirtymää.

Kuinka agentteja voi kuvata? (1/6)

Agentteja voidaan muodostaa laskusääntöjen avulla. Seuraavassa esitetään viisi erilaista tapaa muodostaa uusia agentteja

(Action Prefixing) Jos haluamme, että uusi agenttimme käyttäytyy kuten E sillä erotuksella, että se ensiksi suorittaa tapahtuman α, voimme määritellä

$$F ::= \alpha.E$$

Tälle agentille pätee myös triviaalisti seuraava ominaisuus: $\alpha.E \xrightarrow{\alpha} E$

Kuinka agentteja voi kuvata? (2/6)

(Choice) Jos haluamme uuden agentin käyttäytyvän kuten jompikumpi kahdesta aiemmin määritellystä agentista, voimme kirjoittaa

$$C ::= A + B$$

Epädeterministisen valinnan laskusäännöt voidaan jakaa kahteen osaan:

1. Jos $A \xrightarrow{\alpha} A'$, niin pätee $A + B \xrightarrow{\alpha} A'$
2. Jos taasen $B \xrightarrow{\alpha} B'$, niin pätee $A + B \xrightarrow{\alpha} B'$

Kuinka agentteja voi kuvata? (3/6)

(Parallel Composition) Jos kaksi agenttia kytketään rinnan, syntyy uusi agentti, joka merkitään seuraavasti:

$$L ::= M||N$$

Rinnankytkennän laskusäännöt ovat seuraavat:

1. Jos $M \xrightarrow{\alpha} M'$, ja N ei osallistu ko. tapahtumaan (eli tapahtumaa $\overline{\alpha}$ ei voi suorittaa), pätee $M||N \xrightarrow{\alpha} M'||N$.

2. Jos $N \xrightarrow{\alpha} N'$ ja M ei osallistu ko. tapahtumaan (eli tapahtumaa $\overline{\alpha}$ ei voi suorittaa), on vastaavasti $M||N \xrightarrow{\alpha} M||N'$

3. Jos taas $M \xrightarrow{\alpha} M'$, sekä $N \xrightarrow{\overline{\alpha}} N'$, niin pätee $M||N \xrightarrow{\tau} M'||N'$. Tätä tapahtumaa kutsutaan synkronoinniksi.

Kuinka agentteja voi kuvata? (4/6)

(Restriction) Jos haluamme konstruoida agentin, joka on identtinen toiseen agenttiin sillä erotuksella, ettemme salli tiettyjä tapahtumia, kirjoitamme

$$T = S \setminus \mathsf{Restr}$$

missä Restr on niiden tapahtumien joukko, joita emme salli T:n suorittavan. (Huom. Jos tietty tapahtuma on joukossa Restr, emme salli ko. tapahtuman komplementtitapahtumaakaan suoritettavan).

Tämän operaattorin laskusääntö on seuraava: Jos pätee $S \xrightarrow{\alpha} S'$ ja $\alpha, \overline{\alpha} \notin \mathsf{Restr}$, voimme kirjoittaa $S \setminus \mathsf{Restr} \xrightarrow{\alpha} S' \setminus \mathsf{Restr}$.

Kuinka agentteja voi kuvata? (5/6)

(Relabeling) Jos agentin J tapahtumat nimetään uudelleen käyttäen hyväksi funktiota $m : Act \to Act : m(\overline{\alpha}) = \overline{m(\alpha)}$, saadaan tulokseksi uusi agentti K, joka merkitään

$$K ::= J[m]$$

(Em. ehto funktiolle m tarkoittaa yksinkertaisesti sitä, että jos m kuvaa α tapahtuman β tapahtumaksi, niin se myös kuvaa $\overline{\alpha}$ tapahtuman $\overline{\beta}$ tapahtumaksi.)

Uudelleennimeämisoperaattoria sovelletaan käyttäen seuraavaa laskusääntöä:

Jos $E \xrightarrow{\alpha} E'$, niin pätee myös $E[m] \xrightarrow{m(\alpha)} E'[m]$.

Kuinka agentteja voi kuvata? (6/6)

- Edellä mainitut operaatiot ovat käyttökelpoisia yhdisteltäessä yksinkertaisia agentteja yhä monimutkaisemmiksi.

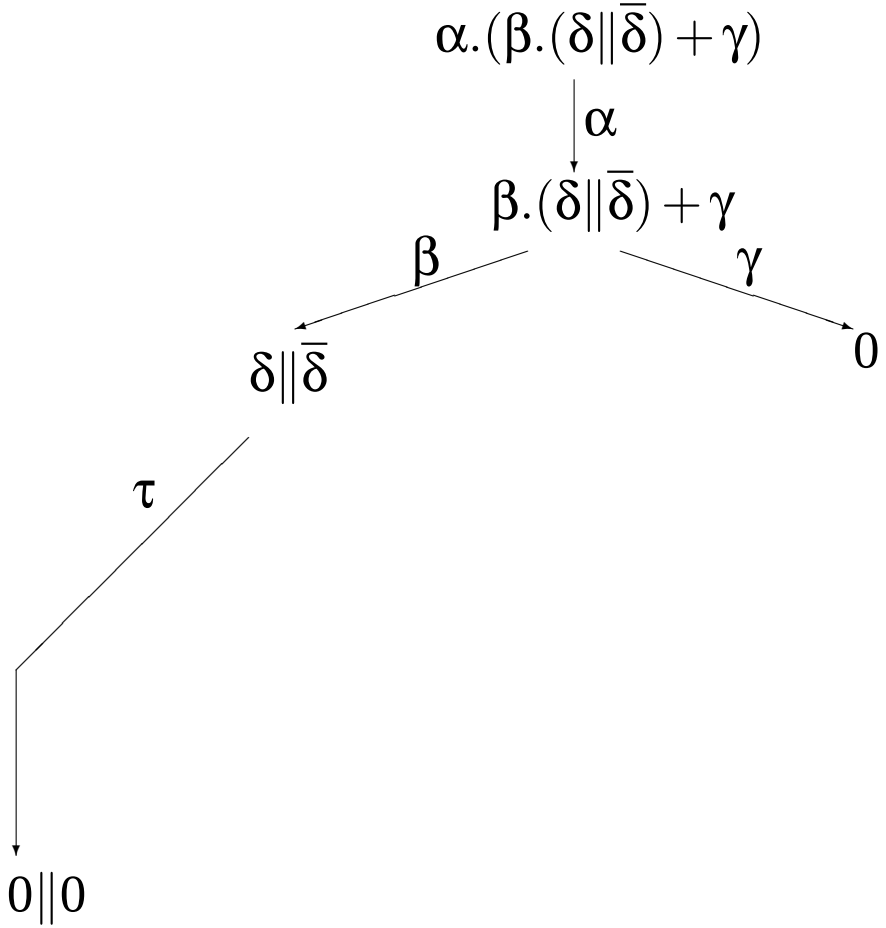
- Jos kyseessä on edellä mainittujen laskutoimitusten ketjutus, voi olla helpompi kuvata proseduuri prosessialgebrallisella yhtälöryhmällä (jotka ovat yleensä helpompia kuin differentiaaliyhtälöt. . .).

- Jos $E ::= \alpha.E'$, ja $E' ::= \beta.\tau.E + \gamma.\tau.E$, voimme yhdistää nämä yhtälöt muotoon

$$E ::= \alpha.(\beta.\tau.E + \gamma.\tau.E)$$

Seuraavalla sivulla on kuvattu agentin $\alpha.(\beta.(\delta||\overline{\delta}) + \gamma)$ käännös graafiseksi. Käännöksessä on sovellettu äskeisiä laskusääntöjä. Kuhunkin tilaan on merkitty siinä mahdollisia toimintoja kuvaava prosessialgebran lauseke.

Teemu Tynjälä


$\alpha.(\beta.(\delta \| \bar{\delta}) + \gamma)$
α
$\beta.(\delta \| \bar{\delta}) + \gamma$
β
γ
$\delta \| \bar{\delta}$
0
τ
$0 \| 0$

Prosessialgebralliset ekvivalenssit (1/3)

- CCS määrittelee useita eri ekvivalenssityyppejä, joihin nojaten on mahdollista verrata kahta agenttia.

- Tutustutaan jälkiekvivalenssiin *(trace equivalence),* hylkäysekvivalenssiin *(failure equivalence)* ja simulaatioekvivalenssiin *(simulation equivalence)* .

(Trace equivalence) Olkoot kaksi agenttia P ja Q. Jos P:n ja Q:n äärellisten suoritusten (ne tapahtumaketjut, mitä agentti pystyy suorittamaan) joukot ovat samat, kyseiset agentit ovat jälkiekvivalentit.

Esim. $\alpha.(\beta+\gamma)$ agentin jäljet ovat $\{\langle\alpha,\beta\rangle,\langle\alpha,\gamma\rangle\}$. Entäs agentin $\alpha.\beta+\alpha.(\beta+\gamma)$ jäljet? Nekin ovat $\{\langle\alpha,\beta\rangle,\langle\alpha,\gamma\rangle\}$. Täten agentit ovat jälkiekvivalentit.

Prosessialgebralliset ekvivalenssit (2/3)

(Failure Equivalence) Agentin P hylkäys *(failure)* on pari $\langle \sigma, X \rangle$, mikä tarkoittaa sitä, että lähtien liikkeelle agentista P, suorittamalla σ pääsemme johonkin toiseen agenttiin Q, jossa mikään X:n tapahtumista ei ole mahdollinen. Jos kahden agentin kaikki hylkäykset ovat samat, nämä kaksi agenttia ovat hylkäysekvivalentit.

Agentin $\alpha.(\beta + \gamma)$ hylkäykset ovat $\langle \langle \alpha \rangle, \{\alpha\} \rangle$, $\langle \langle \alpha, \beta \rangle, \{\alpha, \beta, \gamma\} \rangle$ ja $\langle \langle \alpha, \gamma \rangle, \{\alpha, \beta, \gamma\} \rangle$.

Vastaavasti agentin $\alpha.\beta + \alpha.(\beta + \gamma)$ hylkäysjoukko muodostuu alkioista $\langle \langle \alpha \rangle, \{\alpha, \gamma\} \rangle$, $\langle \langle \alpha \rangle, \{\alpha\} \rangle$, $\langle \langle \alpha, \beta \rangle, \{\alpha, \beta, \gamma\} \rangle$ ja $\langle \langle \alpha, \gamma \rangle, \{\alpha, \beta, \gamma\} \rangle$. Kuten voimme huomata, nämä kaksi agenttia *eivät* ole hylkäysekvivalentit.

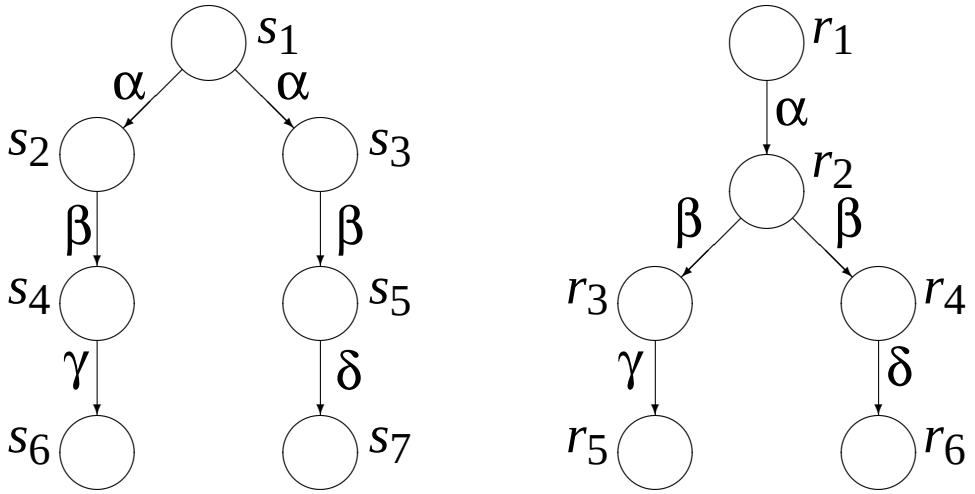
Prosessialgebralliset ekvivalenssit (3/3)

(Simulation Equivalence) Agentti F simuloi agenttia E, jos on olemassa binäärinen relaatio $\mathcal{R} \subseteq S \times S$ kaikkien agenttien yli, joka toteuttaa

1. $E \; \mathcal{R} \; F$ eli $(E, F) \in \mathcal{R}$
2. Jos $E' \; \mathcal{R} \; F'$ ja $E' \xrightarrow{\alpha} E''$, on olemassa F'' siten, että $F' \xrightarrow{\alpha} F''$ ja lisäksi $E'' \; \mathcal{R} \; F''$

Seuraavalla kalvolla on laskettu kahden agentin välinen simulaatio ja todettu agentit simulaatioekvivalenteiksi.

Simulaatioekvivalenssin laskuesimerkki



Simulaatiorelaatio $\mathcal{R}$ on $\{(s_1, r_1), (s_2, r_2), (s_4, r_3), (s_6, r_5), (s_3, r_2), (s_5, r_4), (s_7, r_6)\}$.

Yhteenveto

- Prosessialgebra on tekstuaalinen tapa kuvata agenttien käyttäytymistä.
- Agentteja voidaan yhdistää ja määritellä osa kerrallaan soveltamalla prosessialgebran operaatioita.
- Agentteja voi myös kuvata prosessialgebran yhtälöryhmillä.
- Kahta agenttia voidaan tehokkaasti vertailla käyttäen erilaisia ekvivalenssirelaatioita. Tässä käsiteltiin jälkiekvivalenssia, hylkäysekvivalenssia, ja simulaatioekvivalenssia. (Muitakin on olemassa!)

Teemu Tynjälä