1 PAIKKA/TRANSITIO-VERKOT

- Yleistetään edellä esitettyä ideaa seuraavasti:
  - Jokaiseen paikkaan voidaan selvästi laittaa useampia täpliä.
  - Toisaalta paikoista voi lähteä useampia kaaria samaan transitioon ja päinvastoin.
  - Täplät voidaan nyt tulkita resursseiksi.

©Johan Lilius 1997, Keijo Heljanko 1998

Verkko

Definition 1 Kolmikkoa $N =< S, T; F >$ kutsutaan *verkoksi*, joss:

1. S ja T ovat erilliset $(S \cap T = \emptyset)$, niin sanottujen $S-$ ja $T-$elementtien joukot, ja
2. $F \subseteq (S \times T) \cup (T \times S)$ on binäärinen *vuorelaatio*

- Annetulle alkioille $a \in S \cup T$, joukkoa ${}^\bullet a = \{a' \mid a'Fa\}$ kutsutaan alkion a *esijoukoksi*, ja vastaavasti joukkoa $a^\bullet = \{a' \mid aFa'\}$ alkion a *jälkijoukoksi*.

©Johan Lilius 1997, Keijo Heljanko 1998

Paikka/Transitio-verkko

Definition 2 6-monikko $N =< S, T; F, K, M_0, W >$ on paikka/transitio- (P/T-) verkko (engl. *Place/Transition Net*, saks. *Stellen/Transition-Netz*), joss:

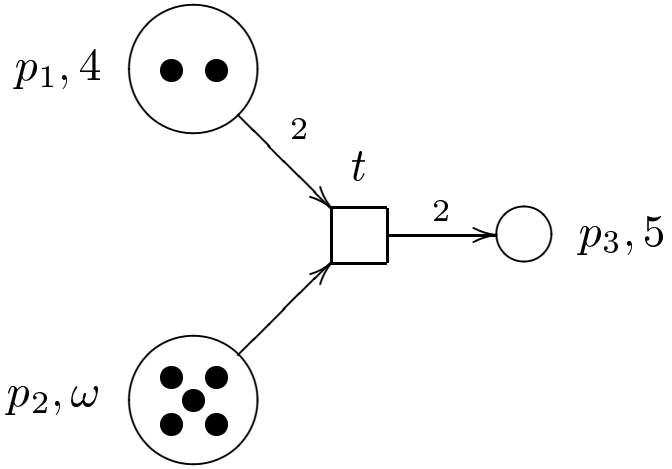
1. $< S, T; F >$ on äärellinen Petri-verkko; joukkojen S ja T alkioita kutsutaan vastaavasti *paikoiksi* ja *transitioiksi*.
2. funktio $K : S \to (\mathbb{N} \uplus \{\omega\})$ antaa kullekin paikalle *kapasiteetin* (capacity), mahdollisesti rajoittamattoman, mitä merkitään symbolilla $\{\omega\}$,
3. funktio $W : F \to (\mathbb{N} \backslash \{0\})$ antaa kunkin kaaren *kaaripainon* (arc weight), ja
4. $M_0 \in \mathsf{M_N}$ on verkon *alkumerkintä* (initial marking), missä:

$$\mathsf{M_N} = \{M : S \to \mathbb{N} \uplus \{\omega\} \mid \forall s \in S : M(s) \leq K(s)\},$$

on verkon N, *merkintöjen* (markings) joukko.

©Johan Lilius 1997, Keijo Heljanko 1998

Paikka/Transitio-verkko



1. $< S, T; F >=< \{p_1, p_2, p_3\}, \{t\}; \{(p_1, t), (p_2, t), (t, p_3)\} >$
2. $K = \{p_1 \mapsto 4, p_2 \mapsto \omega, p_3 \mapsto 5\}$
3. $W = \{(p_1, t) \mapsto 2, (p_2, t) \mapsto 1, (t, p_3) \mapsto 2\}$
4. $M_0 = \{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 0\}$

©Johan Lilius 1997, Keijo Heljanko 1998

©Johan Lilius 1997, Keijo Heljanko 1998

Laukeamissääntö

Definition 3 Olkoon $N =< S, T; F, K, M_0, W >$ P/T-verkko. Olkoon M verkon N merkintä.

1. Transitio t on *M-virittynyt* (merkitään $M[t >$) joss

$$\begin{aligned} \forall s \in {}^\bullet t \quad &: \quad M(s) \geq W(s,t) \wedge \\ \forall s \in t^\bullet \quad &: \quad M(s) \leq K(s) - W(t,s). \end{aligned}$$

   (Laukeamissuehto)

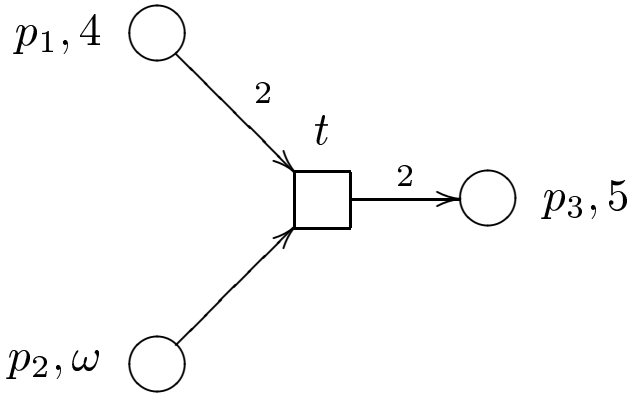
2. M-virittynyt transitio t voi *laueta* tuottaen *seuraajamerkinnän* M' (merkitään $M[t > M'$), jolle pätee

$$\forall s \in S : M'(s) = M(s) - W(s,t) + W(t,s).$$

   (Laukeamissääntö)

3. Merkitään $[M_0 >= \{M \in \mathsf{M_N} \mid M_0[>^* M\}$, merkinnästä M_0 äärellisellä transitiojonolla saavutettavien merkintöjen joukkoa.

Laukeamissääntö



Alkumerkinnällä:

- $M_0 = \{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 0\}$ transitio t on vireessä.
- $M_0 = \{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 5\}$ transitio t ei ole vireessä.
- $M_0 = \{p_1 \mapsto 1, p_2 \mapsto 5, p_3 \mapsto 0\}$ transitio t ei ole vireessä.

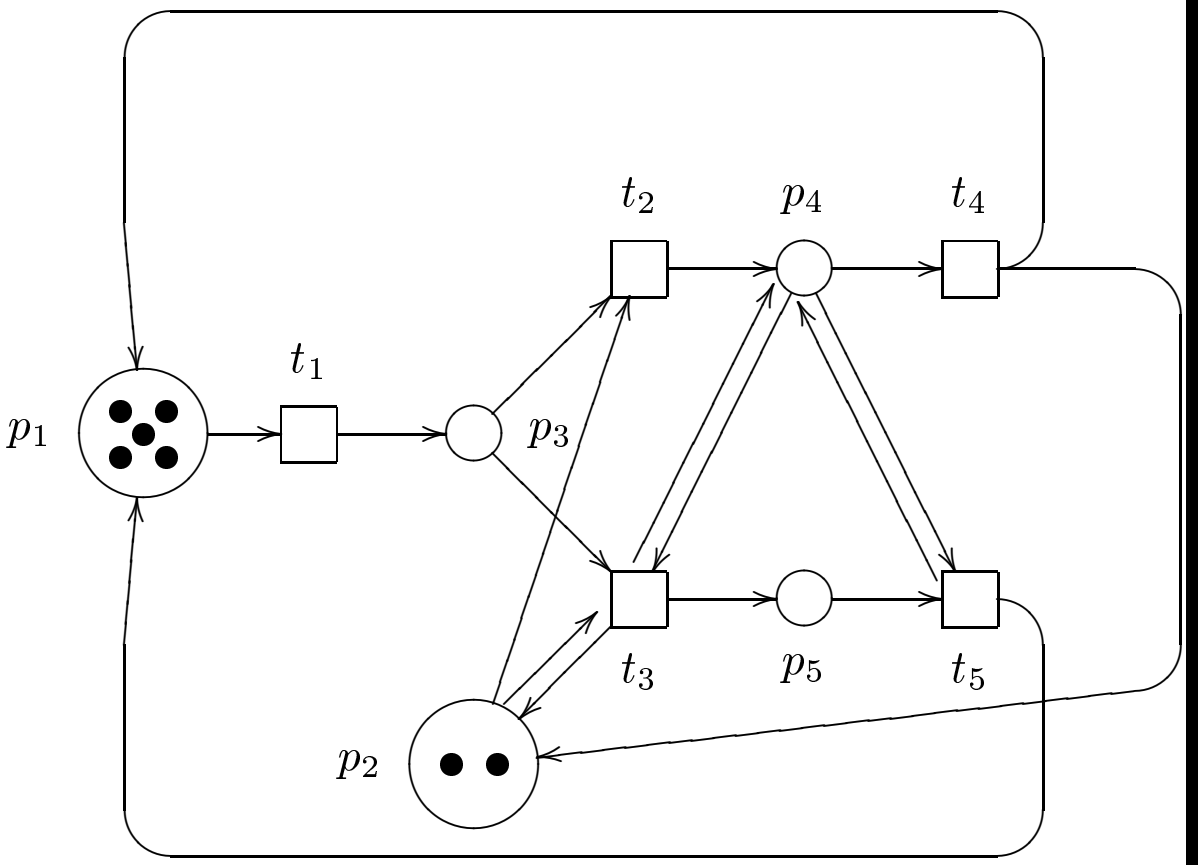
©Johan Lilius 1997, Keijo Heljanko 1998

Esimerkki: Moniprosessorijärjestelmä

- 5 prosessoria
- 3 yhteistä muistia
- 2 väylää
- Mallitus: Tarkastellaan vain järjestelmän resurssikäyttäytymistä
- Prosessorien tilat:
  - aktiivinen
  - odottaa väylän vapautumista
  - odottaa muistin vapautumista
  - lukee/kirjoittaa muistia
- Tehdään prosessorin tiloista paikkoja.
- Lisäksi tarvitaan paikka väyläresursseille.

©Johan Lilius 1997, Keijo Heljanko 1998

Esimerkki: Moniprosessorijärjestelmä



- t_1: saantipyyntö,
- p_3: prosessori odottaa väylän vapautumista,
- p_4: prosessori on saanut muistin käyttöönsä,
- p_5: prosessori jonottaa käytössä olevaa muistia,
- t_5: jonotus loppuu (prosessori pääsee muistiin käsiksi)
- t_2, t_3: muistin valinta, jo käytössä oleva (t_3) tai vapaa (t_2),
- t_4: muistilla ei enää pyyntöjä.
- Paikkojen kapasiteetti ω.

©Johan Lilius 1997, Keijo Heljanko 1998

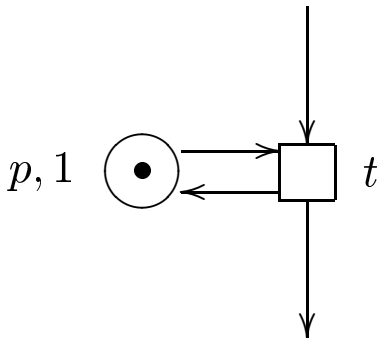
Esimerkki: Moniprosessorijärjestelmä

- Mallista nähdään esimerkiksi:
  - Kun yksi prosessori on varannut muistin voidaan toisen väylän kautta viedä saantipyyntöjä muistiin jonottamaan.
  - Jos kaksi prosessoria valitsee vapaan muistin jäävät muut odottamaan.
  - Kaikille verkon merkinnöille M pätee:
    1. $M(p_1) + M(p_3) + M(p_4) + M(p_5) = 5$, sekä
    2. $M(p_2) + M(p_4) = 2$
  - Ko. ominaisuudet ovat verkon *invariantteja*.
- HUOM:
  - Ei tiedetä prosessorin, väylän, ja muistin *identiteettiä* (asiaan palataan myöhemmin).
  - Maalliin voidaan lisätä mielivaltainen määrä prosessoreita.
  - Entä väyliä?
  - Malli ei ota kantaa muistien määrään.
  - Ajattelemalla transitioiden laukeamisille todennäköisyydet voidaan tarkastella järjestelmän kvantitatiivista käyttäytymistä (asiaan palataan myöhemmin): esim. kuinka monta prosessoria on keskimäärin odottamassa paikassa p_5

©Johan Lilius 1997, Keijo Heljanko 1998

Tiukka Silmukka

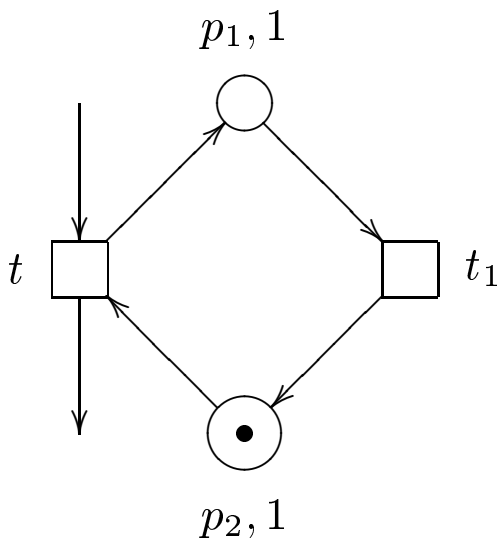
- Joissain tilanteissa on luontevaa käyttää seuraavaan kaltaista rakennetta verkon osana:



  (vrt. moniprosessorijärjestelmä)
- t ei laukeamisehdon mukaan ole vireessä.
- Laukeamisehtoa voidaan muuttaa muotoon:

$$\begin{aligned} \forall s \in {}^\bullet t \quad &: \quad M(s) \geq W(s,t) \wedge \\ \forall s \in t^\bullet \quad &: \quad M(s) - W(s,t) + W(t,s) \leq K(s). \end{aligned}$$

- Tai verkko voidaan muuntaa muotoon:



©Johan Lilius 1997, Keijo Heljanko 1998

Saavutettavuusgraafi

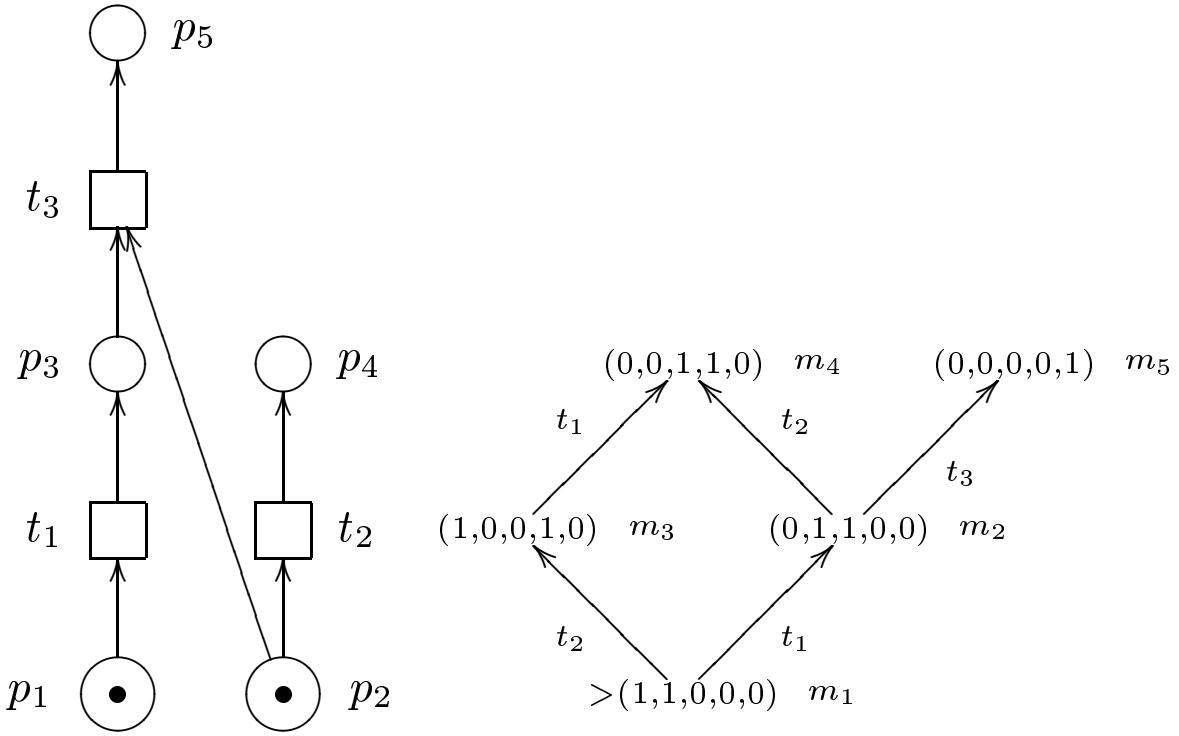
Definition 4 P/T-verkon $N =< S, T; F, K, M_0, W >$ *saavutettavuusgraafi* (reachability graph) on juurellinen, kaarimerkitty graafi $SG =< V; A, r >$, jossa $r = M_0, V = [M_0 >$, ja

$$A = \{< M, t, M' > \mid M \in V \wedge M[t > M'\}$$

- Käytännöllisesti katsoen kaikki analysointityökalut ja menetelmät edellyttävät saavutettavuusgraafin laskentaa.
- Saavutettavuusgraafi sallii rinnakkaisen ja hajautetun järjestelmän tarkastelun tavallisena automaattina.
- Vastaavasti saavutettavuusgraafista ei enää suoraan voida nähdä, mitkä toiminnot ovat riippumattomia ja rinnakkaisia.

©Johan Lilius 1997, Keijo Heljanko 1998

Saavutettavuusgraafi



- Kaikki kapasiteetit ja kaaripainot $= 1$.
- Usein käytetään myös notaatiota $\{p_1, p_2\}$ notaation $(1, 1, 0, 0, 0)$ sijaan.
- Riippumattomat transition t_1 ja t_2 *lomitettu*.

©Johan Lilius 1997, Keijo Heljanko 1998

Saavutettavuusgraafin generointi

- Oletetaan annetuksi funktiot:
  1. $enabled(M)$: palauttaa merkinnässä M viireessä olevien transitioiden joukon, sekä
  2. $fire(t, M)$ palauttaa uuden merkinnän, joka saadaan laukaisemalla transitio t merkinnässä M.

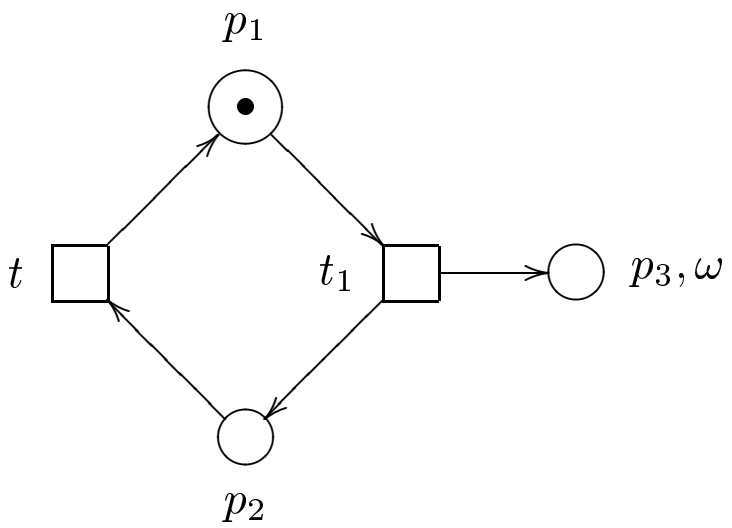
Reachability-Graph($N :< S, T; F, K, M_0, W >$)

```
1  G : graph; Markings : stack;
2  G ← ∅;
3  Markings.push(M_0);
4  while Markings ≠ ∅
5  do M ← pop(Markings);
6      for t in enabled(M)
7      do M' ← fire(t, M);
8          if M' ∉ G
9              then push(M', Markings);
10         G ← G ∪ {< M, t, M' >};
```

©Johan Lilius 1997, Keijo Heljanko 1998

Äärelliset saavutettavuusgraafit

- Ääretön saavutettavuusgraafi:



- Saavutettavuusgraafi:

$$(1,0,0) \xrightarrow{t_1} (0,1,1) \xrightarrow{t} (1,0,1) \xrightarrow{t_1} (0,1,2) \quad \cdots$$

- Paikan p_3 merkintä kasvaa rajatta.
- Voidaanko verkon rajoittuneisuus päätellä?

©Johan Lilius 1997, Keijo Heljanko 1998

ÄÄRELLISET SAAVUTETTAVUUSGRAAFIT

Definition 5 Annettu P/T-verkko N on *k-rajoitettu* (k-bounded), joss

$$\forall M \in [M_0 >, s \in S : M(s) \leq k.$$

P/T-verkko on *rajoitettu* (bounded), joss on olemassa $k \geq 0$, jolle N on k-rajoitettu. Muuten verkko on *rajoittamaton* (unbounded).

- Verkon rajoittuneisuus voidaan ratkaista muuntamalla saavutettavuusgraafin generointialgoritmia:
  - Merkintä M *peittää* merkinnän M', joss $M \geq M'$.
  - Ennenkuin merkintä M lisätään graafiin, tarkastetaan onko siihen johtavalla polulla merkintää M' s.e. $M \geq M'$. Jos tällainen löytyy lisätään graafin merkinnän M sijaan merkintä M'', jolle:
    - $M''(s) = \omega$, jos $M(s) > M'(s)$,
    - $M''(s) = M(s)$ muuten.

©Johan Lilius 1997, Keijo Heljanko 1998

Peittävyyspuu

```
Coverability-Tree(N)
 1  T : Tree; Markings : Stack;
 2  T ← ∅;
 3  Markings ← M_0;
 4  while Markings ≠ ∅
 5  do M ← pop(Markings);
 6      for t in enabled(M)
 7      do M' ← fire(t, M);
 8          M'' ← M;
 9          while (M'' ≠ nil) ∧ ¬(M'' ≤ M')
10          do M'' ← T.pred(M'');
11          if M'' ≠ nil
12            then M' ← M' + (M' − M'') · ω;
13          T.addLeaf(M, t, M');
14          M'' ← M;
15          while (M'' ≠ nil) ∧ (M'' ≠ M')
16          do M'' ← T.pred(M'');
17          if M'' = nil
18            then push(M', Markings);
```

©Johan Lilius 1997, Keijo Heljanko 1998

- Peittävyys:
  - $M \geq M'$ joss $\forall s \in S : M(s) \geq M'(s)$, eli

$$\begin{aligned} (1,1,0) &\geq (1,0,0) \\ (1,0,0) &\not\geq (0,0,1) \end{aligned}$$

  - HUOM: $M \not\geq M'$ ei implikoi $M < M'$.
- Oletetaan annetuksi funktiot
  - $T.pred(M)$ joka palauttaa puussa T M:ää edeltävän solmun, sekä
  - $T.addleaf(M, t, M')$ joka lisää puuhun uuden lehden M' ja lisää siihen kaaren t solmusta M.
- Rivillä 8-11 kuljetaan puussa takaperin, kunnes joko saavutaan juureen, tai löytyy merkintä M'' jonka M' peittää.
- Rivillä 13 esim.:

$$\begin{aligned} (1,1,0) &= (1,1,0) + ((1,1,0) - (1,1,0)) \cdot \omega \\ &= (1,1,0) + (0,0,0) \cdot \omega \\ &= (1,1,0) + (0 \cdot \omega, 0 \cdot \omega, 0 \cdot \omega) \\ &= (1,1,0) + (0,0,0) \\ (1,\omega,0) &= (1,1,0) + ((1,1,0) - (1,0,0)) \cdot \omega \\ &= (1,1,0) + (0,1,0) \cdot \omega \\ &= (1,1,0) + (0 \cdot \omega, 1 \cdot \omega, 0 \cdot \omega) \\ &= (1,1,0) + (0,\omega,0) \end{aligned}$$

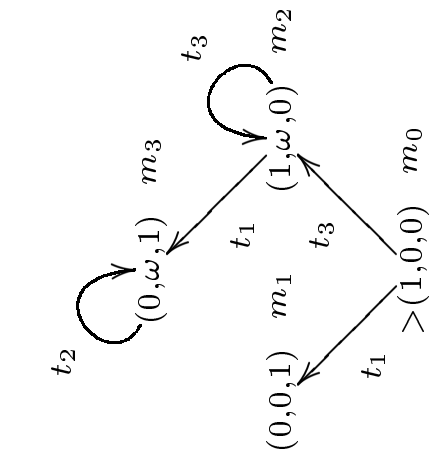
©Johan Lilius 1997, Keijo Heljanko 1998

koska:

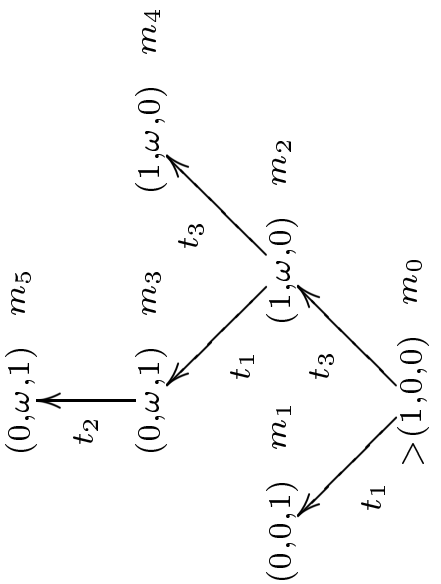
- $\forall n \in \mathbb{N} : \omega > n, \omega + n = \omega - n = \omega, \omega \geq \omega, n \cdot \omega = \omega \ \ (n > 0), 0 \cdot \omega = 0.$
- Rivillä 15 lisätään puuhun merkintä M' uutena lehtenä, vaikka ko. merkinät olisikin jo löytynyt.
- Rivillä 16-19 kuljetaan puussa takaperin, kunnes joko löydetään juuri generoitu merkintä, tai saavutaan juureen.
- Jälkimmäisestä voidaan päätellä, että merkinnässä ei ole aikaisemmin käyty, joten se voidaan lisätä pinoon (rivit 20-22).
- Peittävyyspuusta tehdään peittävyysgraafi yhdistämällä samanarvoiset solmut. Tämä voidaan tietysti jo tehdä suoraan puuta generoidessa, jolloin kuitenkin rivillä 9-11 ja 17-19 oleva silmukat muuttuvat graafin läpikäynniksi.
- HUOM: Peittävyysgraafi ei ratkaise äärettömän P/T-verkon saavutettavuusongelmaa, koska peittävän merkinnän löytäminen ei välttämättä takaa juuri etsityn merkinnän saavuttamista.

©Johan Lilius 1997, Keijo Heljanko 1998

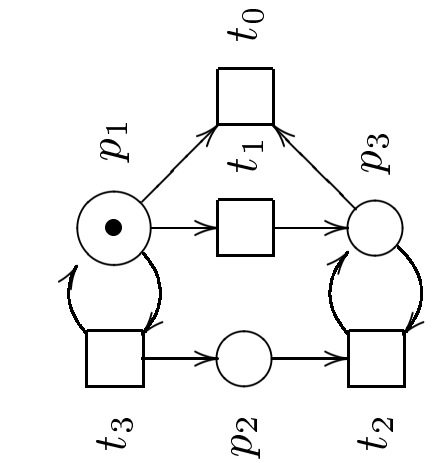
ESIMERKKI



- Peittävyysgraafi



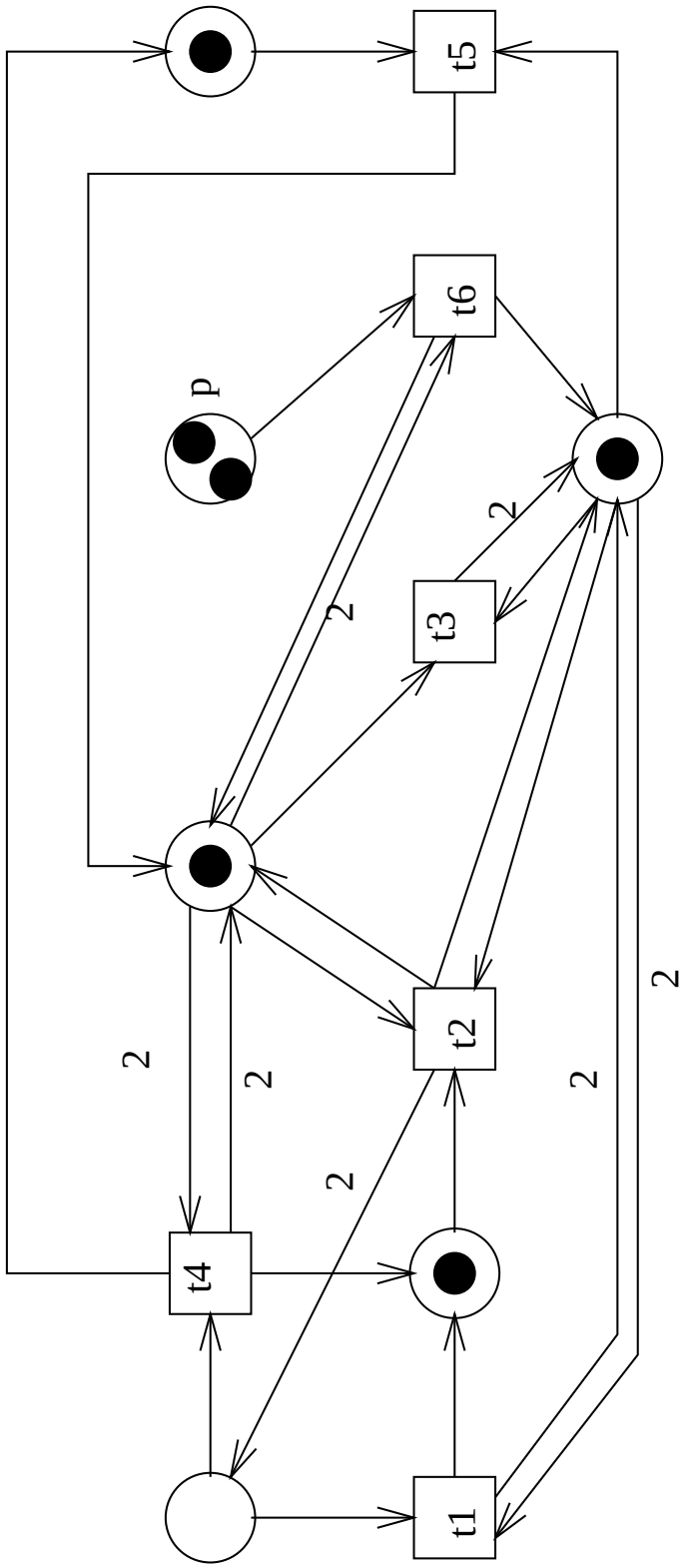
- Peittävyyspuu



- Rajoittamaton verkko

©Johan Lilius 1997, Keijo Heljanko 1998

KOMPLEKSISUUS



©Johan Lilius 1997, Keijo Heljanko 1998

Kompleksisuus

- Kuvan verkko on rajoitettu kaikille alkumerkinnöille, joissa paikkan p voidaan lisätä mielivaltainen määrä täpliä.
- Alkumerkinnän koko on siis $M_0(p) + 4$, ja verkon koko $M_0(p) + 48$.
- Olkoon $Z(k)$ M_0:sta saavutettavissa olevan merkinnän maksimaalinen koko kun $M_0(p) = k$. Tälle funktiolle pätee:

$$\begin{aligned} Z(k) &= 2 \cdot f(k) + 2 \\ f(0) &= 2 \\ f(k+1) &= f(k) \cdot 2^{f(k)} \end{aligned}$$

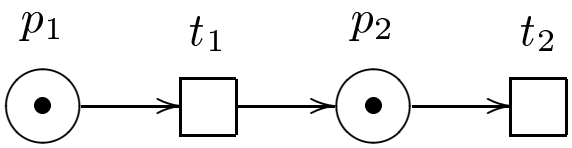
- $Z(0) = 6, Z(1) = 18, Z(2) = 4098, Z(3) = 2^{2060} + 2, \ldots$.
- Saavutettavien merkintöjen määrä: $M(0) = 30, M(1) = 427, \ldots$.
- Yleisessä tapauksessa aikakompleksisuus voi siis olla yliexponentiaalinen: funktio kasvaa nopeammin kuin mikään exponetiaalifunktio (vrt. Ackermannin funktio).
- Kuitenkin 1-rajoitetulle verkolle

$$|[M_0 > | \leq 2^S$$

©Johan Lilius 1997, Keijo Heljanko 1998

KONTAKTI

- Verkossa, jossa paikoilla on äärellinen kapasiteetti voi syntyä merkintä, jossa on *ns. kontakti*.
- Tällöin transition ei ole vireessä, koska jokin sen jälkipaikoista on täynnä (vrt. tiukka silmukka).



- Ilmiön käytännön tulkinta voisi olla seuraavanlainen: jos
  - $p_1 =$ on rahaa,
  - $t_1 =$ nosta luuri,
  - $p_2 =$ soita,
  - $t_2 =$ lopeta puhelu,

  niin ylläoleva verkko kuvaa tilannetta jossa sinulla on rahaa joten voisit soittaa, mutta joku muu on juuri soittamassa, eli puhelin ei ole vapaa.

©Johan Lilius 1997, Keijo Heljanko 1998

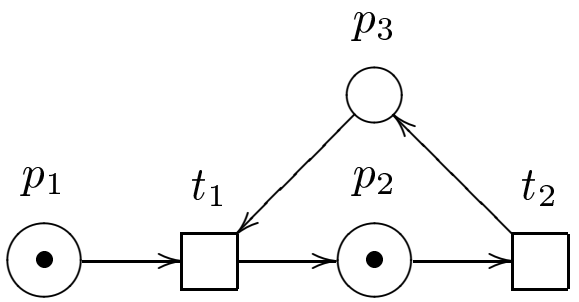
KONTAKTI

- Verkko voidaan kuitenkin muuntaa kontaktittomaksi *komplementtikonstruktion* avulla:

  Jokaista *äärellisellä kapasiteetilla* varustettua paikkaa s kohti verkkon lisätään uusi paikka s', jolla on:

  1. kaari $< s', t >$ transitioon t, joss alkuperäisessä verkossa on kaari tFs $(W(s', t) = W(t, s))$,
  2. kaari $< t, s' >$ transitioista t, joss alkuperäisessä verkossa on kaari sFt $(W(t, s') = W(s, t))$, ja
  3. alkumerkintä määritellään toteuttamaan $M_0(s) + M_0(s') = K(s)$ kaikille paikka-komplementtipaikka pareille s, s'.

- Esimerkistä saadaan:

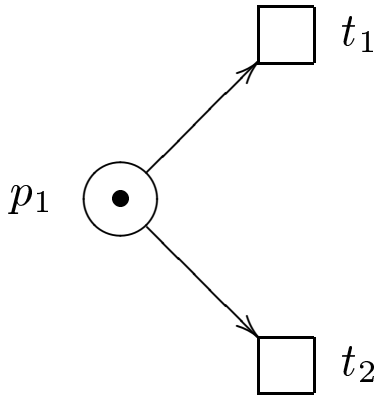


©Johan Lilius 1997, Keijo Heljanko 1998

©Johan Lilius 1997, Keijo Heljanko 1998

Sekaannus

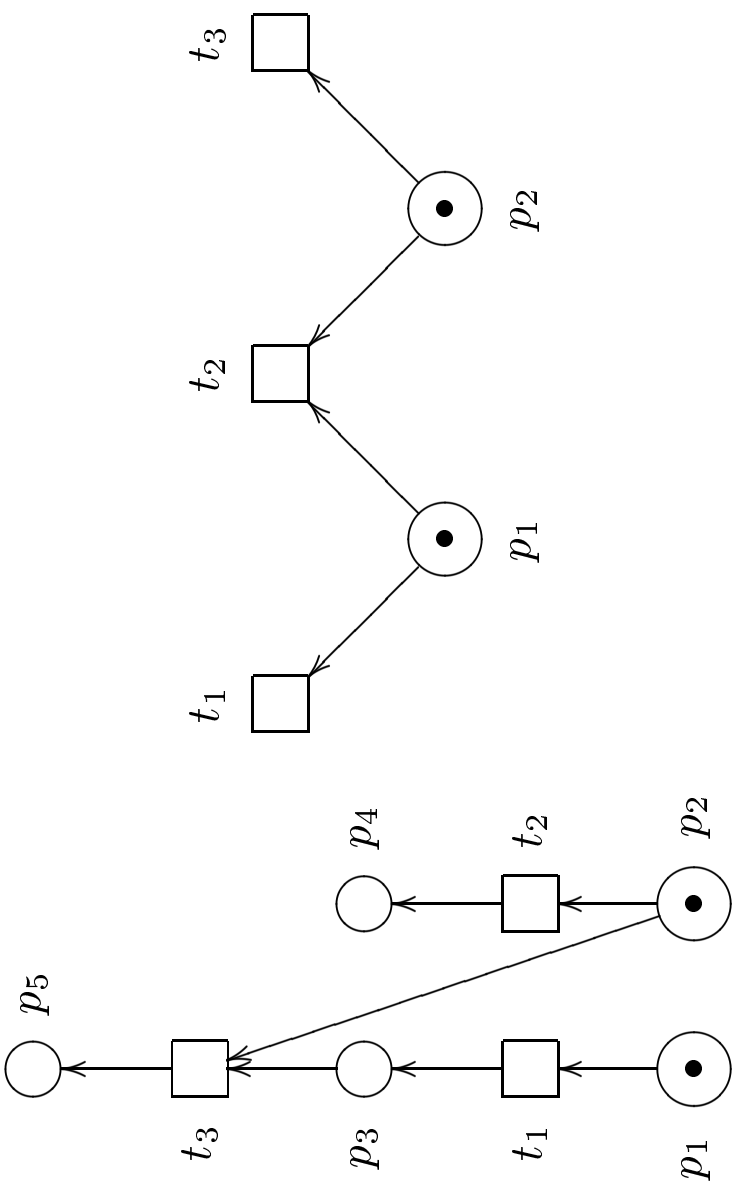
- Kaksi transitiota ovat riippumattomia joss niillä ei ole yhteisiä esi- eikä jälkipaikkoja. Tällöin ne voivat laueta mielivaltaisessa järjestyksessä, tai rinnakkain. Transitiot ovat *samanaikaiset* (concurrent)
- *Kilpatilanne* (conflict) syntyy, kun useampi transitio jakaa esipaikkoja:



  Tällöin vain toinen transitioista voi laueta. Kilpatilanne kuvaa järjestelmän epä-deterministisyyttä.
- Tilannetta jossa esiintyy sekä rinnakkaisuutta että kilpatilanteita kutsutaan nimellä *sekaannus* (confusion).

SEKAANNUS

- Epäsymmetrinen ja symmetrinen sekaannus:



©Johan Lilius 1997, Keijo Heljanko 1998

Sekaannus

- Sekaannus syntyy vasemmassa tapauksessa siitä, että ei ole selvää esiintyykö t_2 ja t_3 välillä kilpatilannetta:
  - Oletetaan että järjestelmää tarkailee kaksi havainnoitsijaa H_1, H_2, jotka näkevät vain yhden transition laukeamisen kerrallaan.
  - H_1 havaitsee $\{p_1, p_2\}[t_1; t_2 > \{p_3, p_4\}$
  - H_2 havaitsee $\{p_1, p_2\}[t_2; t_1 > \{p_3, p_4\}$
  - H_1 havaitsee, että t_2 ja t_3 ovat kilpatilanteessa joka ratkaistaan t_2:n hyväksi.
  - H_2 ei havaitse kilpatilannetta.
- Vastaavasti oikeassa tapauksessa on epäselvyyttä siitä kumpi kilpatilanne ratkaistaan ensin:
  - H_1 havaitsee $\{p_1, p_2\}[t_1; t_3 >$
  - H_2 havaitsee $\{p_1, p_2\}[t_3; t_1 >$
  - H_1:n mielestä t_1:n ja t_2:n välinen kilpatilanne ratkaistaan t_1:n hyväksi.
  - H_2:n mielestä t_2:n ja t_3:n välinen kilpatilanne ratkaistaan t_3:n hyväksi.
- Sekaannus aiheuttaa ongelmia:
  1. kun yritetään päätellä onko järjestelmän käytös *reilua*, sekä
  2. kun yritetään kehittää heuristiikkoja joiden avulla saavutettavuusgraafin kokoa voitasiin pienentää.

©Johan Lilius 1997, Keijo Heljanko 1998