T-79.179 **Kevät 2002**
Rinnakkaiset ja hajautetut digitaaliset järjestelmät
Laskuharjoituksen 4 vastaukset
22.2.2002

1. Olkoon PH tehtävässä annettu verkko. $PH \models \phi$, jos ja vain jos LTL-kaava ϕ pätee verkon PH kaikille äärettömille suorituksille.

   a)

   $$PH \models \Box\neg((E(ph_1) \wedge E(ph_2)) \vee (E(ph_2) \wedge E(ph_3)) \vee (E(ph_3) \wedge E(ph_1)))$$

   Kaksi naapuria ei voi siis syödä samaan aikaan. Sama Marian LTL-kaavana:

   ```
   []!((1,2 subset place eating)
    || (2,3 subset place eating)
    || (3,1 subset place eating));
   ```

   b) Tehdään vastaväite,että ph_1 ole koskaan syömässä:

   $$PH \not\models \Box\neg(E(ph_1))$$

   Vastaväite on epätosi $\Rightarrow$ ph_1 on jossain tilassa syömässä. Sama Marian LTL-kaavana:

   ```
   []!(1 subset place eating);
   ```

   Huomaa, että $\Diamond$-operaattori vaatii, että kysytty ominaisuus pätee kaikilla alkutilasta lähtevillä poluilla, joten:

   $$PH \not\models \Diamond E(ph_1)$$

   c) Filosofi ph_1 voi nääntyä nälkään, jos filosofi ei kaikista tiloista lopulta pääse syömään. (Eli jottei ph_1 nääntyisi, tulee hänen päästä äärettömän monta kertaa syömään.)

   $$PH \not\models \Box\Diamond E(ph_1)$$

   Tämä kaava ei päde mm. lukkiuma-tilaan päätyvillä poluilla. Sama Marian LTL-kaavana:

   ```
   []<>(1 subset place eating);
   ```

   d) Kaava pätee, koska kaikissa tiloissa joissa filosofi ph_1 on syömässä ovat haarukat f_1 ja f_2 varattuina.

   $$PH \models \Box(E(ph_1) \Rightarrow \neg(F(f_1) \vee F(f_2)))$$

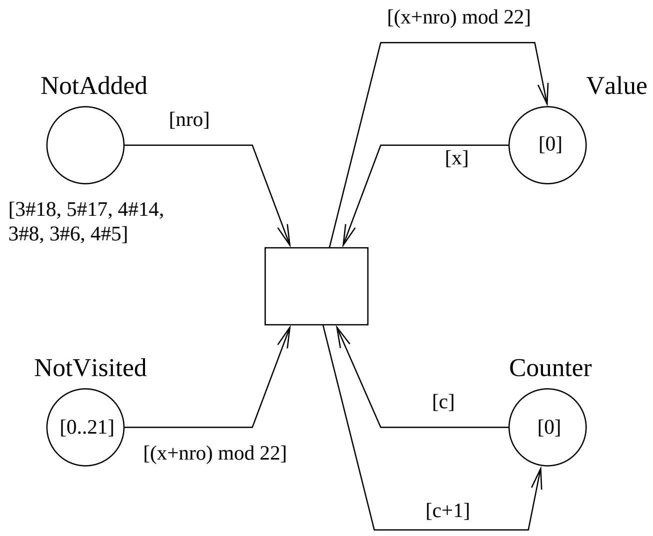
   ```
   [](1 subset place eating => !(1 subset place forks || 2 subset place forks));
   ```

   e) Kaava ei päde, koska järjestelmän deadlock tilassa kaikki haarukat ovat varattuina, mutta filosofi ph_1 ei ole syömässä.

   $$PH \not\models \Box(\neg(F(f_1) \vee F(f_2)) \Rightarrow E(ph_1))$$

   ```
   []!((1 subset place forks || 2 subset place forks) => 1 subset place eating)
   ```

2. a) Tehtävän algebrallinen verkko on esitetty alla:



b) Algebrallista verkkoa vastaava Maria-kuvaus alla:

```
typedef unsigned(0..22) Int;
place NotAdded Int : 3#18,5#17,4#14,3#8,3#6,4#5;
place NotVisited Int : Int a (a <= 21) : a;
place Value Int : 0;
place Counter Int : 0;

trans add
  in {
    place NotAdded : nro;
    place NotVisited : (x+nro)%22;
    place Value : x;
    place Counter : c;
  }
  out {
    Value:      (x+nro)%22;
    Counter:    c+1;
  };
```

c) Verkkoon voidaan helposti lisätä reject-kaava, joka lopettaa saavutettavuusanalyysin, jos järjestelmä tulee tilaan, jossa kaava on tosi. Koska vaatimuksena on löytää ratkaisu 22:lla askeleella, kaava on

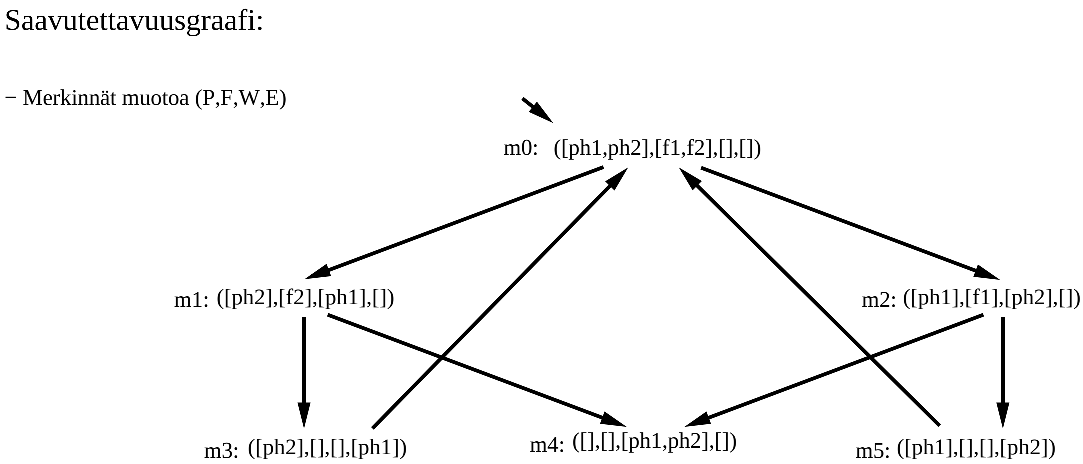
```
reject (is Int 22 equals place Counter) && fatal;
```

Varatun sanan `fatal` evaluointi aiheuttaa saavutettavuusanalyysin keskeyttämisen. Jos `fatal`ia ei käytettäisi, Maria tekisi järjestelmän koko saavutettavuusgraafin ja ilmoittaisi tilat jotka toteuttavat `reject`-ehdon.

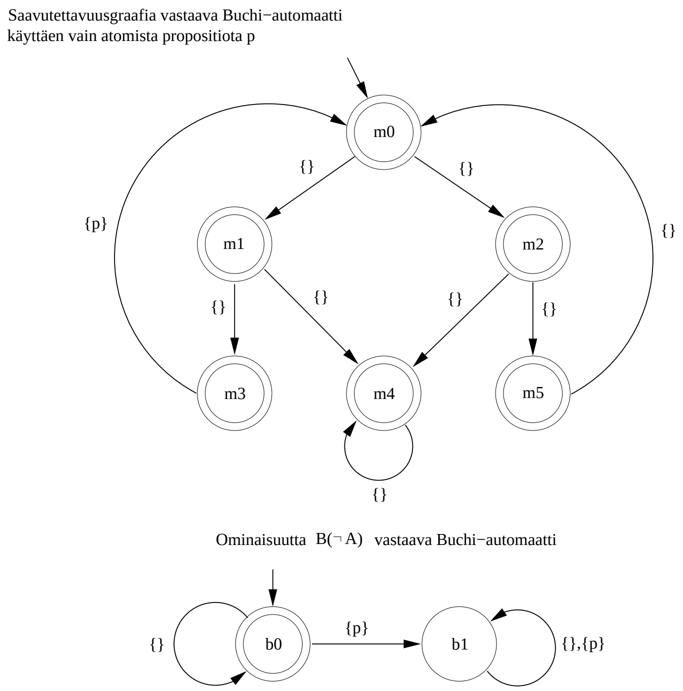
d) Saavutettavuusanalyysi on tämän ongelman ratkaisuun liian voimakas työkalu. Siinä joudutaan muistamaan kaikki jo saavutetut tilat, joita voi olla eksponentiaalinen määrä syötteen kokoon nähden. Käyttämällä hyväksi kombinatoristen ongelmien ratkaisumenetelmiä voidaan kyseinen ongelma ratkaista käyttäen vain lineaarinen määrä tilaa syötteen kokoon nähden.

3. Tämä tehtävä on mukana laskuharjoituksissa demonstraatiotarkoituksessa, sillä sen aihepiiri ei kuulu kurssin suoritusvaatimuksiin.

a) Verkon saavutettavuusgraafi:



b) Saavutettavuusgraafia vastaava Büchi-automaatti $B(G)$ (käyttäen ainoastaan atomista propositiota p) ja kaavan $A = \Diamond p$ negaation ($\Box\neg p$) Büchi-automaatti $B(\neg A)$:



c) Automaattien tulo-automaatti $(B(G) \cap B(\neg A))$ alla. Tulo-automaatilla on ääretön ajo, joka käy hyväksyvässä tilassa äärettömän usein: $(m_0, b_0) \rightarrow (m_2, b_0) \rightarrow (m_5, b_0) \rightarrow (m_0, b_0) \ldots$ Tämän vuoksi järjestelmällä ja kaavan negaatiolla on yhteisiä äärettömiä käytöksiä, joten alkuperäinen kaava $\Diamond p$ ei päde järjestelmässä.

Tilakoneiden B(G) ja B(¬A) tulo