Tik-79.179 **Kevät 2001**
Rinnakkaiset ja hajautetut digitaaliset järjestelmät
Laskuharjoituksen 4 vastaukset
23.2.2001

1. Olkoon PH tehtävässä annettu verkko. $PH \models f$, joss LTL-kaava f on 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 PROD:in verify-kaavana:

   ```
   #verify (henceforth not (((eating >= <.1.>) and (eating >= <.2.>))
                        or ((eating >= <.2.>) and (eating >= <.3.>))
                        or ((eating >= <.3.>) and (eating >= <.1.>))));
   ```

   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 PROD:in verify-kaavana:

   ```
   #verify (henceforth (not (eating >= <.1.>)));
   ```

   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 PROD:in verify-kaavana:

   ```
   #verify (henceforth (eventually (eating >= <.1.>)));
   ```

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

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

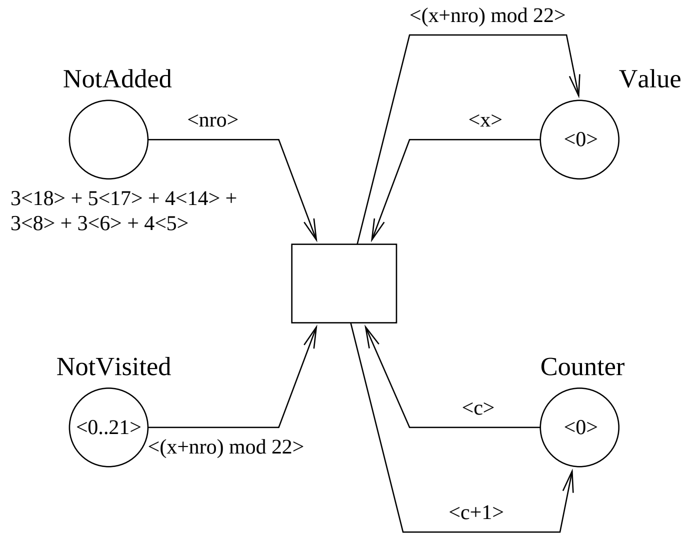
   ```
   #verify (henceforth (eating >= <.1.> implies
                       not(forks >= <.1.> or forks >= <.2.>)));
   ```

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

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

   ```
   #verify (henceforth (not(forks >= <.1.> or forks >= <.2.>) implies
                       eating >= <.1.>));
   ```

2. a) Tehtävän Pr/T-verkko alla:



b) Pr/T -verkkoa vastaava PROD-kuvaus alla:

```
#place NotAdded lo(<.4.>) hi(<.18.>) mk(3<.18.> + 5<.17.> + 4<.14.> +
3<.8.> + 3<.6.> + 4<.5.>)
#place NotVisited lo(<.0.>) hi(<.21.>) mk(<.0..21.>)
#place Value lo(<.0.>) hi(<.21.>) mk(<.0.>)
#place Counter lo(<.0.>) hi(<.22.>) mk(<.0.>)

#trans add
  in {
    NotAdded:   <.nro.>;
    NotVisited: <.(x+nro) % 22.>;
    Value:      <.x.>;
    Counter:    <.c.>;
  }
  out {
    Value:      <.(x+nro) % 22.>;
    Counter:    <.c+1.>;
  }
#endtr
```

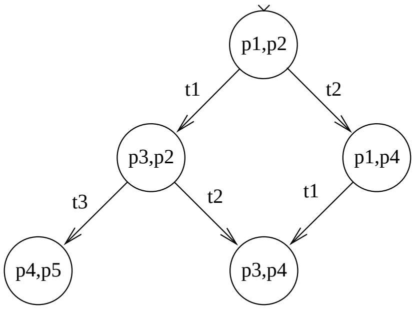
c) Verkkoon voidaan helposti lisätä testeri, joka lopettaa saavutettavuus-analyysin, jos testeri tulee tilaan, joka on reject-tila. Koska counter:in ariteetti on 1, voidaan sitä suoraan käyttää testerinä. Testeri siis laukeaa, jos paikasta Counter löytyy tuppeli `<.22.>`.

```
#tester Counter reject(<.22.>)
```

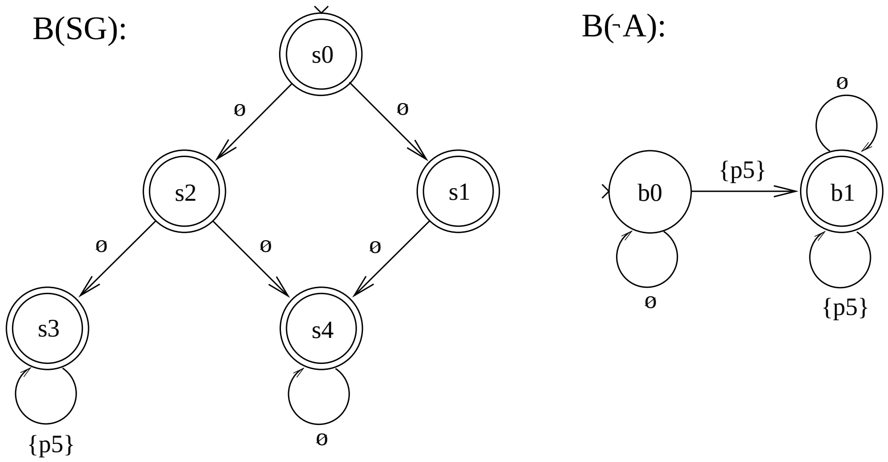
d) Saavutettavuus-analyysi on tämän ongelman ratkaisuun liian voimakas työkalu. Siinä joudutaan muistamaan kaikki jo saavutetut tilat, joita voi olla eksponentiaalinen määrä inputin kokoon nähden. Käyttämällä hyväksi kombinatooristen ongelmien ratkaisumenetelmiä voidaan kyseinen ongelma ratkaista käyttäen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

3. Tämä tehtävä on mukana laskareissa demonstraatio-tarkoituksessa, sen aihepiiri ei kuulu kurssin suoritusvaatimuksiin.

   a) Verkon saavutettavuus-graafi:



   b) Saavutettavuus-graafia vastaava verkon Büchi-automaatti $B(SG)$ (käyttäen ainoastaan atomista propositiota p5) ja kaavan negaation ($\Diamond p5$) Büchi-automaatti $B(\neg A)$:



   c) Automaattien tulo-automaatti ($B(SG) \cap B(\neg A)$) alla. Tulo-automaatilla on ääretön ajo, joka käy hyväksyvässä tilassa äärettömän usein: $(s_0, b_0) \rightarrow (s_2, b_0) \rightarrow (s_3, b_0) \rightarrow (s_3, b_1) \rightarrow (s_3, b_1) \rightarrow (s_3, b_1) \ldots$ Tämän vuoksi järjestelmällä ja kaavan negaatiolla on yhteisiä äärettömiä käytöksiä, joten alkuperäinen kaava $\Box \neg p_5$ ei päde järjestelmässä.