Tik-79.231 **Autumn 2000**
Parallel and distributed digital systems
Answers to Tutorial 4
Oct 27, 2000

1. Let PH be the net given in the problem statement. $PH \models f$, iff the LTL formula f holds for all inifinite executions in the net PH.

   a)

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

   Two neighbours cannot dine at the same time, written as a PROD verify formula:

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

   b) Make teh counter claim that $ph_(1)$ never dines:

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

   The counter claim is untrue $\Rightarrow$ ph_1 dines in some state, written as a PROD verify formula:

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

   N.B.: The $\Diamond$ operater requires that the property hold for all traces from the initial state, so we will have:

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

   c) The philosopher $ph_(1)$ may starve if, whichever the state he is in, the philosopher cannot get to dine eventually. (In other words, so as ph_1 not to starve, he has to get to dine infinitely many times.)

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

   This formula does not hold in the traces ending in a deadlock, written as a PROD verify formula:

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

   d) The formula holds because in all states where the philosopher ph_1 is dining the forks f_1 and f_2 are also taken.

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

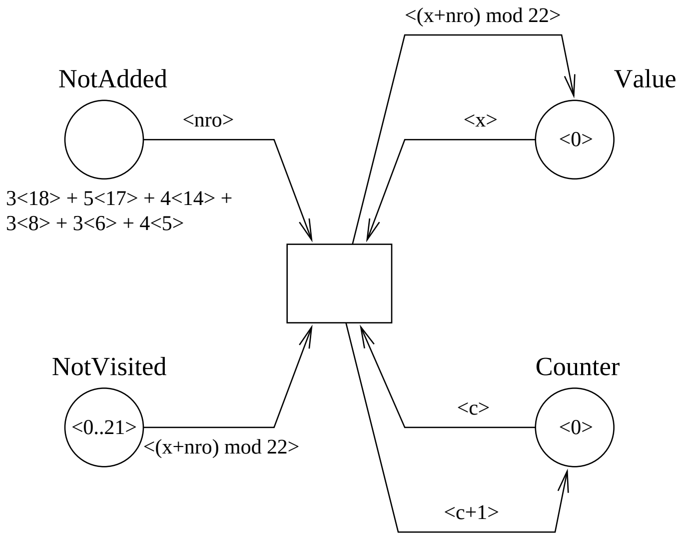
```
#verify (henceforth (eating >= <.1.> implies
                     not(forks >= <.1.> or forks >= <.2.>)));
```

   e) The formula does not hold since in a system deadlock state all forks are taken but the philosopher ph_1 is not dining.

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

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

2. a) The Pr/T net solution is given below:



b) The PROD description is given below:

```
#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
```

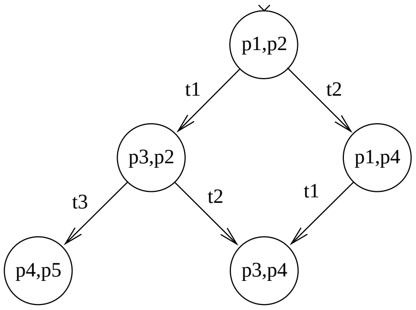
c) A tester can easily be added to the net. The effect is to stop the reachability analysis if the tester is in a state which is a reject state. Since the arity of the counter is 1 it can be used as a tester. The tester fires if the tuple `<.22.>` is found in the place `Counter`.

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

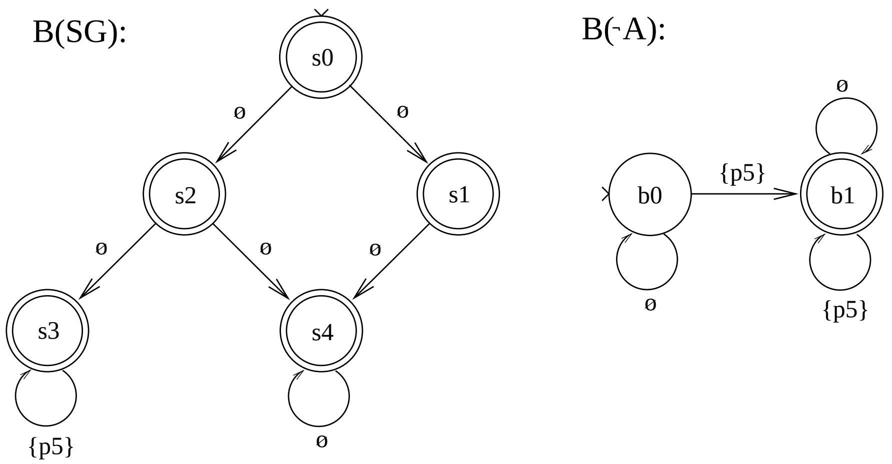
d) To solve this problem reachability analysis is too powerful a tool. In reachability analysis, all aready reached states are stored, and their number can be exponential to the size of the input. Using the solution methods for combinatorial problems we can solve using a linear time w.r.t. the size of the input.

3. Being beyond the scope of the course, this example is used for demonstation purposes only.

   a) Reachability graph of the net:



   b) Büchi automaton corresponding to the reachability graph $B(SG)$ (using only the atomic proposition $p5$ and the Büchi automaton $B(\neg A)$ if the formula negation $(\Diamond p5)$:



   c) The product automaton of the automata $(B(SG) \cap B(\neg A))$ is give below. The product automaton has an infinite run visiting the accepting state infinitely often: $(s_0, b_0) \to (s_2, b_0) \to (s_3, b_0) \to (s_3, b_1) \to (s_3, b_1) \to (s_3, b_1) \ldots$ This causes the system and the negation of the formula having shared infinite behaviour, so the original formula $\Box \neg p_5$ does not hold in the system.