Formalne metode u oblikovanju sustava

3. domaća zadaća - ProMeLa i Spin

Zagreb, svibanj 2020.

Sadržaj

Izjava **2**

Napomena **2**

1. dio **3**
Zadatak a) . . . 3
Zadatak b) . . . 4
Zadatak c) . . . 5
Zadatak d) . . . 5
Zadatak e) . . . 5
Zadatak f) . . . 6

2. dio **7**
Zadatak 1. . . . 7
Zadatak 2. . . . 7
Zadatak 3. . . . 7
Zadatak 4. . . . 8
Zadatak 5. . . . 8
Zadatak 6. . . . 8
Zadatak 7. . . . 8
Zadatak 8. . . . 8
Zadatak 9. . . . 9
Zadatak 10. . . . 9
Zadatak 11. . . . 10

3. dio **11**
Zadatak 1. . . . 11
Zadatak 2. . . . 11
Zadatak 3. . . . 11
Zadatak 4. . . . 12

4. dio **13**
Zadatak a) . . . 14
Zadatak b) . . . 15

Dodatak A - ProMeLa predložak **16**

Izjava

Zadaci priloženi uz ovu domaću zadaću pripadaju njihovim vlasnicima te služe isključivo u edukacijske svrhe. Bilo koja promjena originala je isključivo radi estetske i funkcionalne prirode i ne mijenja smisao informacije. Također, takve promjene ne primijenjuje nikakvu drukčiju licencu niti se smatraju intelektualnim vlasništvom uređivača.

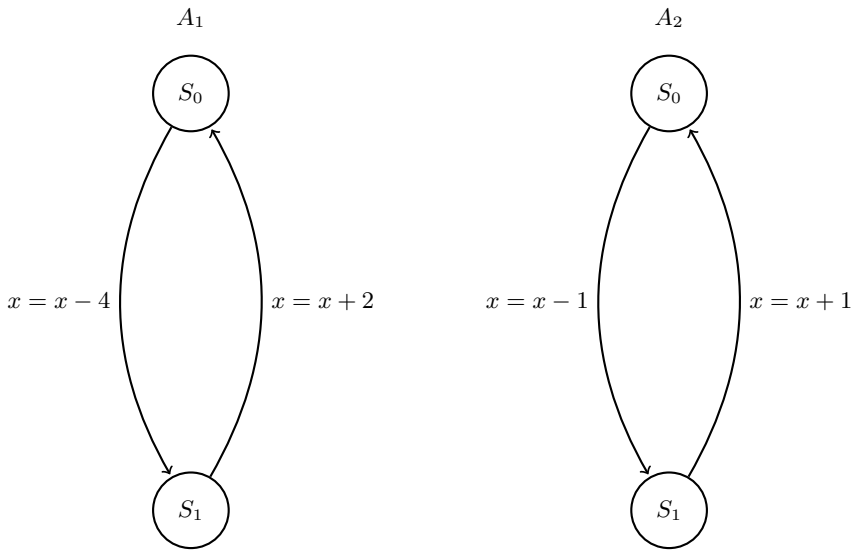
Uz ovu datoteku obično dolaze i primjerci kôda, koji su zaštićeni licencom:

Napomena

Iako u uputama piše da se koriste `.prm` datoteke, često će se odgovarati s `.pml` datotekama. Ovo vas ne bi trebalo zbuniti, jer su datoteke sadržajem iste, a ekstenzija je drukčija samo zato što je uz nju u Visual Studio Code-u dostupno bojanje teksta, pa je lakše pisati taj kod. Isto tako, kad god u odgovoru piše da se pokreće neki kod, to se vrši u korijenskom direktoriju zadaće (tj. u DZ3).

1. DIO

Zadana su dva konačna diskretna automata A_1 i A_2 prema slici (početna stanja su uvijek S_0, a završna S_1):



Zadatak a)

Detaljno opisati strukturu automata A_1 i A_2 prema definiciji FSA $A = (S, s_0, L, T, F)$ (odrediti elemente svakog od skupova $S, s_0, L, ...$)

Odgovor

Oba automata imaju dva stanja: S_0 i S_1.

Na skici nije eksplicitno označeno ulazno stanje s_0, Ali možemo pretpostaviti da je S_0 zbog upute.

Svaki graf ima 2 labele, L_0 i L_1. Prvi graf ima labele:

- $L_0 : x = x - 4$
- $L_1 : x = x + 2$

Drugi graf ima labele:

- $L_0 : x = x - 1$
- $L_1 : x = x + 1$

Oba automata imaju 2 prijelaza, T_0 i T_1. Za prvi automat oni su:

- $T_0 : S_0(x = x - 4) \to S_1$
- $T_1 : S_1(x = x + 2) \to S_0$

Za drugi automat oni su:

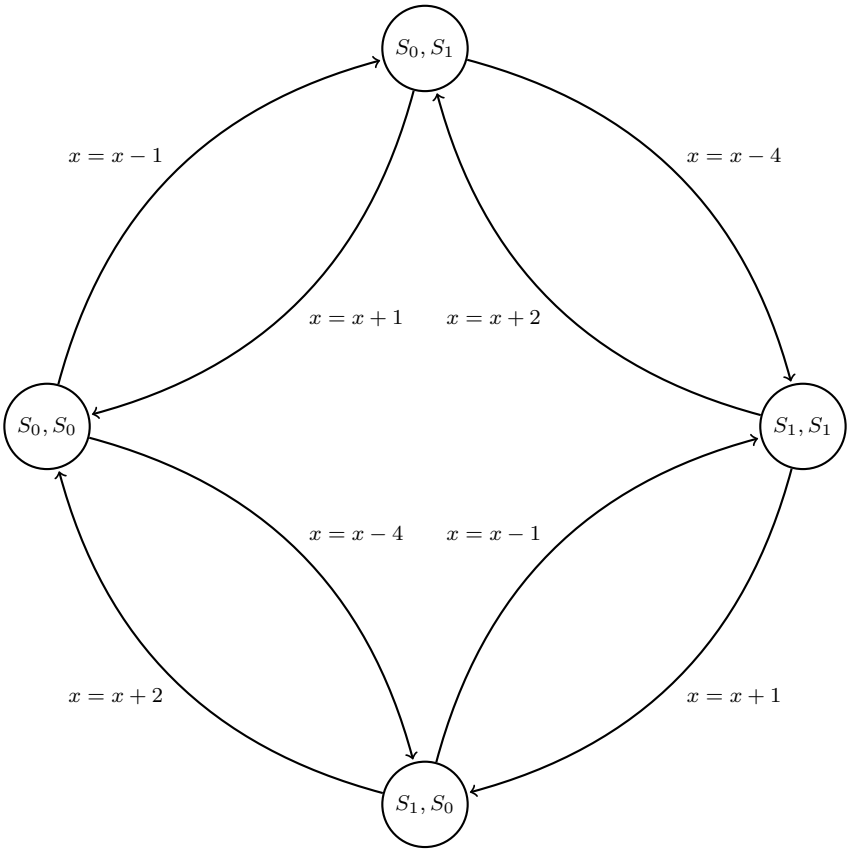
- $T_0 : S_0(x = x - 1) \to S_1$
- $T_1 : S_1(x = x + 1) \to S_0$

Konačno stanje F nije eksplicitno označeno, ali zbog upute zaključujemo da je S_1 kod oba automata.

Zadatak b)

Odrediti asinkroni produkt automata A_1 i A_2 i nacrtati ga.

Odgovor

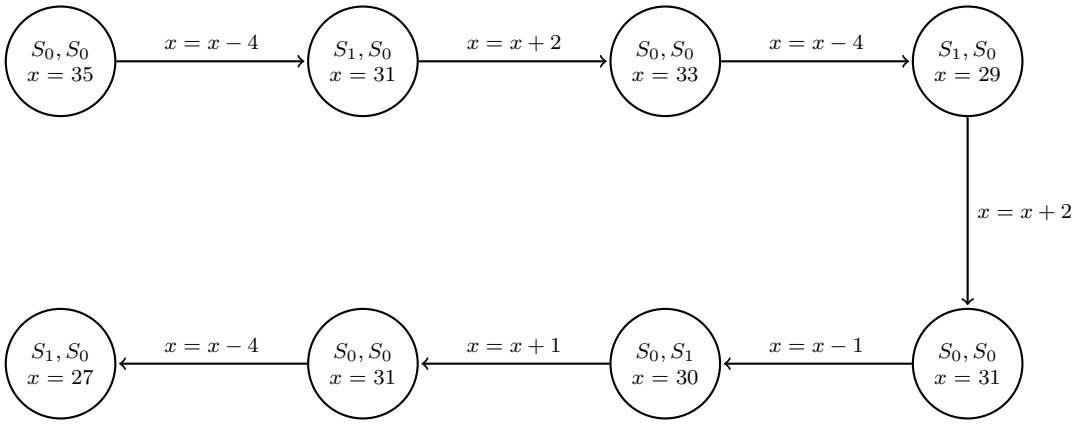


Zadatak c)

Odrediti ekspandirani asinkroni produkt za prvih 5-10 po volji odabranih članova i nacrtati ga.

Odgovor

Uzmimo $x_0 = 35$.



Zadatak d)

Pomoću ekspandiranog produkta odrediti istinitost LTL formule $\Diamond\Box p$ ako je $p \equiv x \leq 0$. Obrazložiti rješenje, posebice komentirati mogućnost rješavanja bez primjene programskih alata.

Odgovor

Formula ispituje je li uvijek globalno vrijedi da je $x \leq 0$. Ovo ne vrijedi jer možemo zapeti u petlji $(S_0, S_0) \to (S_0, S_1) \to (S_0, S_0)$ za neki $x_0 \geq 1$.

Zadatak e)

Pomoću ekspandiranog produkta odrediti istinitost LTL formule $\Diamond p$ ako je $p \equiv x < 0$. Obrazložiti rješenje, posebice komentirati mogućnost rješavanja bez primjene programskih alata.

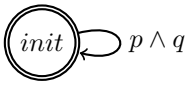
Odgovor

Isto kao i iznad - ne vrijedi uvijek $p \equiv x < 0$ jer možemo zapeti u petlji koja će x održavati pozitivnim ako odaberemo pozitivan početni x.

Zadatak f)

Nacrtati moguću realizaciju Büchi automata za LTL formulu: $\Box(p \wedge q)$.

Odgovor



2. DIO

Zadatak 1.

Instalirajte programski alat Spin. Instalacija se svodi na kopiranje izvršnog programa. Za one koji hoće više, sve instrukcije jezika *ProMeLa* možete pronaći na ovoj poveznici, kao i službene upute ("manual") ovdje.

Zadatak 2.

Uredite automate A_1 i A_2 kao promela procese A i B (vidjeti predložak). Napomena: Promela datoteku nazvati `prezime.prm` (npr. `blaskovic.prm`). Polazeći od zadanog Promela modela koji se sastoji od dva procesa A i B analizirat će se LTL formula $\Diamond p$ gdje je $p \equiv (x \leq 0)$.

Zadatak 3.

Pokrenite simulaciju: `spin -u20 -p -c -g prezime.prm`. Prepišite prvih 12 članova. Pismeno obrazložite istovjetnosti i razlike između ekspandiranog asinkronog produkta iz domaće zadaće i rezultata simulacije.

Odgovor

```
 1:    proc  0 (A:1) ./src/p-2/prezime.pml:8 (state 1)  [(1)]
 2:    proc  1 (B:1) ./src/p-2/prezime.pml:16 (state 1) [(1)]
 3:    proc  1 (B:1) ./src/p-2/prezime.pml:17 (state 2) [x = (x-1)]
               x = 34
 4:    proc  0 (A:1) ./src/p-2/prezime.pml:9 (state 2)  [x = (x-4)]
               x = 30
 5:    proc  0 (A:1) ./src/p-2/prezime.pml:9 (state 3)  [goto S1]
 6:    proc  1 (B:1) ./src/p-2/prezime.pml:17 (state 3) [goto S1]
 7:    proc  0 (A:1) ./src/p-2/prezime.pml:10 (state 4) [x = (x+2)]
               x = 32
 8:    proc  0 (A:1) ./src/p-2/prezime.pml:10 (state 5) [goto S0]
 9:    proc  1 (B:1) ./src/p-2/prezime.pml:18 (state 4) [x = (x+1)]
               x = 33
10:    proc  1 (B:1) ./src/p-2/prezime.pml:18 (state 5) [goto S0]
11:    proc  0 (A:1) ./src/p-2/prezime.pml:9 (state 2)  [x = (x-4)]
               x = 29
12:    proc  1 (B:1) ./src/p-2/prezime.pml:17 (state 2) [x = (x-1)]
               x = 28
```

Ovdje se odvijaju prijelazi u drukčijem redoslijedu (npr. mi smo u prvih 4 iteracija pokretali samo proces 1, ovdje se pokreću naizmjence). Zbog drukčijeg redoslijeda i početne vrijednosti, i konačni rezultat je drukčiji. Jedino što je isto je da će uz dobru vjeru (ako ne bude samo djelovao proces 1) vrijednost konvergirati prema $-\infty$.

Zadatak 4.

Generirajte analizator: `spin -a prezime.prm`.

Zadatak 5.

Prevedite u izvršni oblik npr.: `gcc -o pan pan.c`

Zadatak 6.

Pozovite analizator: `pan -a` ili `./pan -e`. Pismeno obrazložite je li uvjet p zadovoljen.

Odgovor

Uvjet nije zadovoljen:

```
assertion violated  !((x<=0)) (at depth 128)
```

Zadatak 7.

Pokrenite "error trail" opciju (pronalaženje protuprimjera) sa `spin -t -p -c -g prezime.prm`. Postoji li sekvenca u kojoj varijabla x na kraju poprima vrijednost $x \leq 0$? Koliko koraka (*engl. steps*) sadrži?

Odgovor

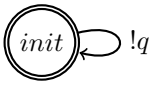
Postoji takva sekvenca. Ona sadrži 129 koraka.

Zadatak 8.

Prepišite instrukcije za Büchi automat koje generira Spin `spin -f '!<>q'`. Nacrtajte pripadni Büchi automat.

Odgovor

```
never  {    /* !<>q */
accept_init:
T0_init:
        do
        :: (! ((q))) -> goto T0_init
        od;
}
```

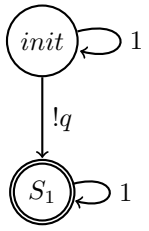


Zadatak 9.

Prepišite instrukcije za Büchi automat koje generira Spin `spin -f '![]q'`. Nacrtajte pripadni Büchi automat.

Odgovor

```
never  {    /* ![]q */
T0_init:
        do
        :: atomic { (! ((q))) -> assert(!(! ((q)))) }
        :: (1) -> goto T0_init
        od;
accept_all:
        skip
}
```

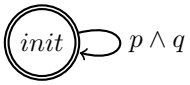


Zadatak 10.

Na isti način koristeći Spin nacrtajte automat iz Vaše domaće zadaće (pitanje `f`)).

Odgovor

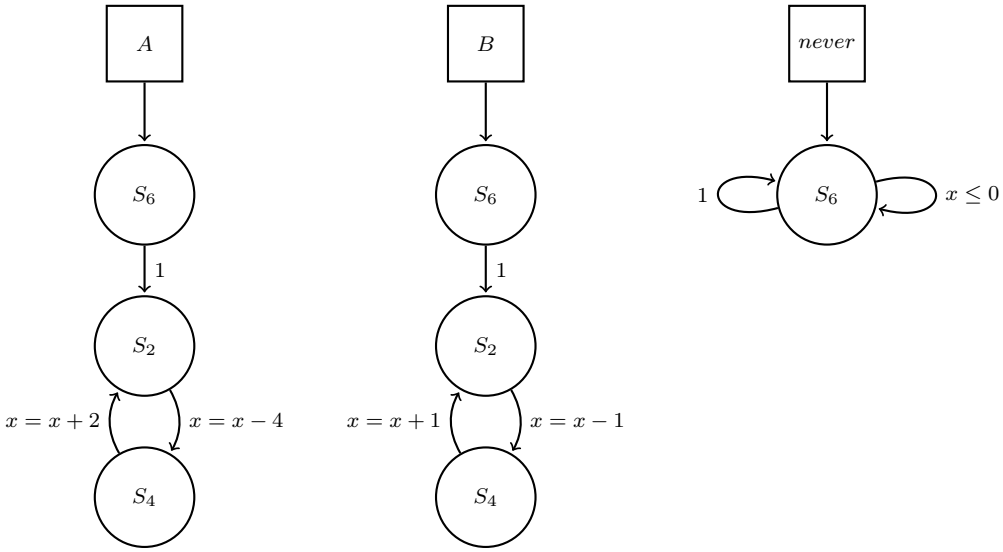
```
never  {    /* [](p && q) */
accept_init:
T0_init:
        do
        :: ((p && q)) -> goto T0_init
        od;
}
```



Zadatak 11.

Generirajte analizator sa `spin -a -o3 prezime.prm`, prevedite te pozovite analizator sa `pan -d`. Precrtajte tako dobivene FSA. Objasnite razlike kao i istovjetnosti prema automatima iz domaće zadaće. Usporedite stanja prema slici automata i stanja dobivena s opcijom `pan -d`. U čemu je razlika?

Odgovor



Prijelazi su isti. Međutim, naredba nam je izbacila i automat za `never`.

3. DIO

Ponovite postupak za $\Diamond\Box p$ (modificirati "*never claim*" `spin -f "<>[]p"` na kraju prezime.prm datoteke).

Zadatak 1.

Postoji li sekvenca u kojoj varijabla x na kraju poprima vrijednost $x \leq 0$?

Odgovor

Pokretanjem `spin -a prezime.prm` utvrđujemo da takva sekvenca postoji.

Zadatak 2.

Koliko koraka (*engl. steps*) sadrži?

Odgovor

Zadnjih par linija ispisa `spin -t -p -c -g ./src/p-3/prezime.pml` nam vraća:

```
136:    proc  0 (A:1) prezime.pml:9 (state 2)    [x = (x-4)]
                x = -2
Never claim moves to line 25    [((x<=0))]
138:    proc  1 (B:1) prezime.pml:18 (state 4)   [x = (x+1)]
                x = -1
  <<<<<START OF CYCLE>>>>>
Never claim moves to line 30    [((x<=0))]
140:    proc  1 (B:1) prezime.pml:17 (state 2)   [x = (x-1)]
                x = -2
142:    proc  1 (B:1) prezime.pml:18 (state 4)   [x = (x+1)]
                x = -1
spin: trail ends after 142 steps
```

Stoga utvrđujemo da se to događa nakon 139 koraka.

Zadatak 3.

Je li moguće problem riješiti bez LTL formule, samo pomoću `assert naredbi`? Obrazložite odgovor.

Odgovor

To nije moguće. Razlog za to je što kad bi koristili `assert`, moramo ga koristiti nakon što je uvjet ispunjen. Ako znamo kad je uvjet ispunjen, koja je svrha validacije?

Zadatak 4.

Je li moguće problem riješiti bez LTL formule, samo pomoću simulacije? Obrazložite odgovor.

Odgovor

To isto nije moguće. Imamo beskonačno puteva izvođenja, i još k tome petlju!

4. DIO

Zadan je Promela model komunikacijskog protokola koji opisuje dio moguće realizacije protokola za preuzimanje datoteka (*engl. download*):

```
mtype = { ini, ack, dreq, data, shutup, quiet, dead };
chan M = [N] of { mtype };
chan W = [N] of { mtype };

active proctype Mproc() {
    W!ini;
    M?ack;

    timeout ->
        if
            :: W!shutup
            :: W!dreq;
                M?data ->
                    do
                        :: W!data
                        :: W!shutup;
                        break
                    od
        fi;

    M?shutup;
    W!quiet;
    M?dead
}
active proctype Wproc() {
    W?ini;
    M!ack;

    do
        :: W?dreq ->
            M!data
        :: W?data ->
            #if 1
            M!data
            #else
            skip
            #endif
        :: W?shutup ->
            M!shutup;
        break
    od;

    W?quiet;
    M!dead
}
```

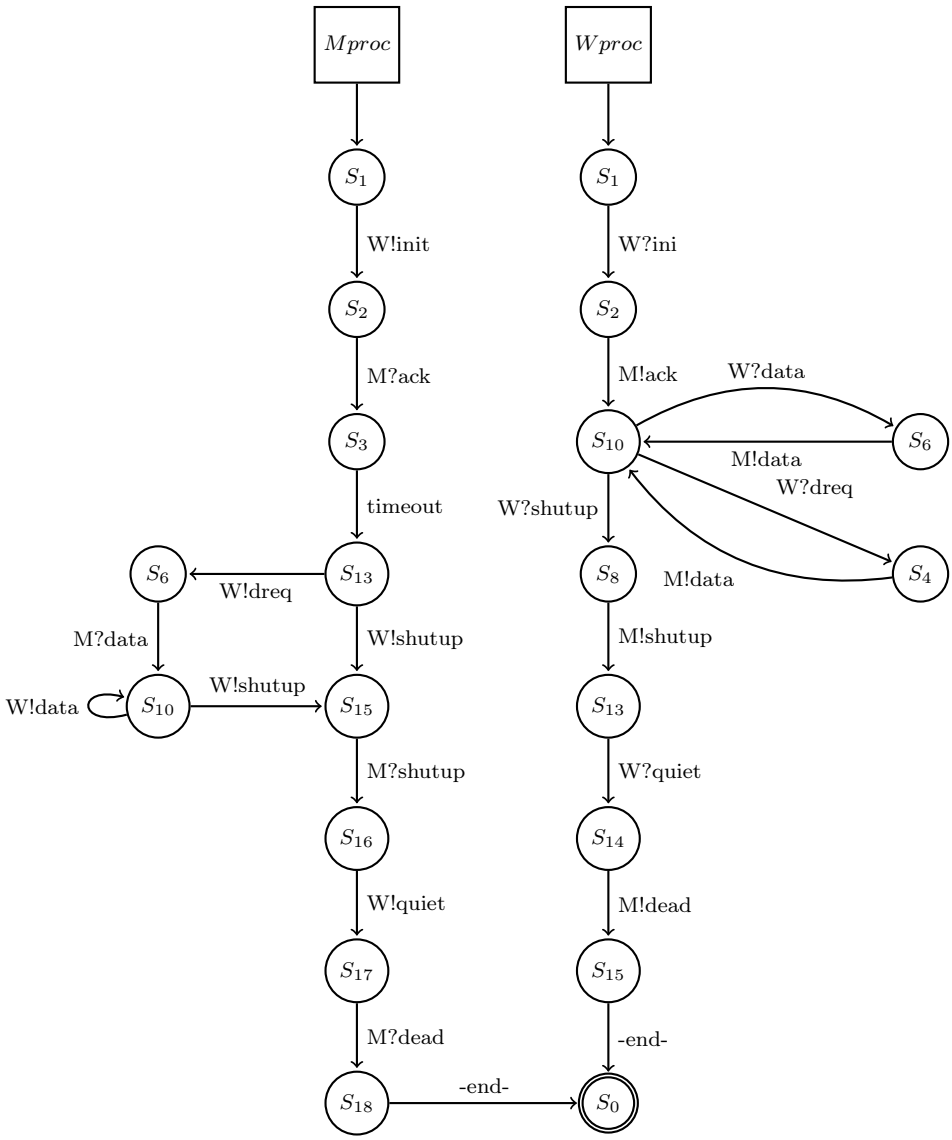
Ako je $N = 16$, pomoću programskog alata `Spin` odredite:

Zadatak a)

Iz kojih stanja nema napretka (nastupa tzv. potpuni zastoj, *engl. deadlock*)?

Odgovor

Prvo moramo skicirati automate:



Kao što vidimo sa slike, potpuni zastoj može nastupiti u S_{10}.

Zadatak b)

Dolazi li protokol u završno stanje?

Odgovor

Da. Ovo možemo provjeriti s `spin -p -c -g ./src/p-4/prezime.pml`. Zadnjih par linija ispisa nam daju:

```
12:    proc  0 (Mproc:1) prezime.pml:24 (state 15)      [M?shutup]
               queue 2 (M):
               queue 1 (W):
 1   W!quiet
13:    proc  0 (Mproc:1) prezime.pml:25 (state 16)      [W!quiet]
               queue 2 (M):
               queue 1 (W): [quiet]
 1   .   W?quiet
14:    proc  1 (Wproc:1) prezime.pml:47 (state 13)      [W?quiet]
               queue 2 (M):
               queue 1 (W):
 2   .   M!dead
15:    proc  1 (Wproc:1) prezime.pml:48 (state 14)      [M!dead]
               queue 2 (M): [dead]
               queue 1 (W):
 2   M?dead
16:    proc  0 (Mproc:1) prezime.pml:26 (state 17)      [M?dead]
               queue 2 (M):
               queue 1 (W):
16:    proc  1 (Wproc:1)           terminates
16:    proc  0 (Mproc:1)       terminates
```

Ovo znači da je moguće doći do konačnog stanja.

Dodatak A - ProMeLa predložak

```
#define N 35
#define p (x <= 0)

int x = N;

active proctype A() {
do
    ...
od;
}

active proctype B() {
do
    ...
od;
}

never { /* LTL formula */
    ...
}
```