PERO PASKOVIĆ
0036509921

KOMUNIKACIJSKI PROTOKOLI
PRVA LABORATORIJSKA VJEŽBA
SPIN/PROMELA

U prvoj laboratorijskoj vježbi sam imao zadatak simulirati komunikaciju 2 predajnika i 2 prijamnika preko mreže.
Uvjeti komunikacije su bili sljedeći:

- predajnik 1 šalje pakete prijamniku 1
- predajnik 2 šalje pakete prijamniku 2
- komunikacija je sinkrona
- predajnik šalje novi paket tek kad primi potvrdu o primitku prethodnog paketa
- vjerojatnost gubitka paketa je 33 %
- vjerojatnost gubitka potvrde je 10 %
- mreža prima i prosljeđuje pakete tako da prvo proslijedi sve pakete koje šalje prvi predajnik čiju poruku primi, a zatim prima i prosljeđuje sve pakete drugog prijamnika
- ne smije biti gomilanja paketa na kanale
- poruka je u obliku 14 paketa

PERO PASKOVIĆ
0036509921

KOMENTIRANJE KODA

Kod je detaljno iskomentiran, isječke ću postaviti i ovdje.

```
init {
	chan pred_ch1 = [0] of {int, int, int, int};
	chan pred_ch2 = [0] of {int, int, int, int};
	chan prij_ch1 = [0] of {int, int, int, int};
	chan prij_ch2 = [0] of {int, int, int, int};

	run Predajnik (1,1,pred_ch1); // predajnik 1 salje pakete prijamniku 1
	run Predajnik (2,2,pred_ch2); // predajnik 2 salje pakete prijamniku 2
	run Mreza (pred_ch1,pred_ch2,prij_ch1,prij_ch2);
	run Prijamnik(prij_ch1);
	run Prijamnik(prij_ch2);
}
```

Ovdje možemo vidjeti proces init gdje su inicijalizirana 4 kanala (pred_ch1 spaja predajnik 1 s mrežom, pred_ch2 spaja predajnik 2 s mrežom, prij_ch1 spaja mrežu s prijamnikom 1, a prij_ch2 spaja mrežu s prijamnikom 2). Sinkronost komunikacije se može vidjeti u brojci 0 koja označava da se na kanalu sprema 0 podataka. Sva četiri podatka koja se šalju kanalom su tipa int.
Također je pokrenuto i 5 procesa. Dva procesa pokreću 2 predajnika, jedan proces pokreće mrežu te dva procesa pokreću 2 prijamnika.

PERO PASKOVIĆ
0036509921

```
proctype Predajnik(int oznaka_predajnik; int oznaka_prijamnik; chan kanal){
	int br_poruke = 0;
	int oznaka_potvrde;
	int vrsta_poruke;

	do
	:: (br_poruke < 14) ->
		kanal! oznaka_predajnik,oznaka_prijamnik,br_poruke,0;
		if
		:: timeout -> kanal! oznaka_predajnik,oznaka_prijamnik,br_poruke,0;
		:: kanal? oznaka_predajnik,oznaka_prijamnik,oznaka_potvrde,vrsta_poruke; br_poruke = oznaka_potvrde;
		fi;
	:: else -> break
	od
}
```

U ovom isječku, koji predstavlja proces Predajnik, možemo vidjeti da dok god ima poruka, tj. dok god je oznaka poruke manja od 14 na kanal se stavlja oznaka pošiljatelja, oznaka primatelja, oznaka poruke te broj 0 što u ovom slučaju označuje da je poruka paket (broj 1 označuje da je poruka potvrda). U slučaju isteka određenog vremena predviđenog za primanje potvrde provodi se retransmisija. Na kanalu prij_ch1 ili prij_ch2 (zavisi od pošiljatelja i primatelja) se primaju podatci o predajniku, prijamniku, rednom broju potvrde te vrsti_poruke (0 za paket, 1 za potvrdu). Također se i u varijablu br_poruke sprema redni broj pristigle potvrde.
Varijabla vrsta_poruke je uvedena ponajviše zbog if-statementa u procesu Mreža da mi je lakše upravljati gubitkom paketa/potvrda.

PERO PASKOVIĆ
0036509921

```
proctype Prijamnik(chan kanal){
	int oznaka_predajnik;
	int oznaka_prijamnik;
	int br_poruke = 0;
	int vrsta_poruke;

	do
		:: (br_poruke < 14) ->
			kanal? oznaka_predajnik,oznaka_prijamnik,br_poruke,vrsta_poruke;
			br_poruke++;
			kanal! oznaka_predajnik,oznaka_prijamnik,br_poruke,1;
		:: else -> break
	od;
}
```

U ovom isječku koji predstavlja proces Prijamnik, možemo vidjeti da dok god je redni broj poruke manji od 14 s kanala se primaju podatci o oznaci predajnika, prijamnika, dohvaća se redni broj poruke koji se uvećava za 1 te se dohvaća oznaka poruke (0 za paket, 1 za potvrdu) nakon toga se na isti kanal stavljaju podatci o predajniku, prijamniku, rednom broju poruke te broj 1 koji označava da je poruka potvrda.

PERO PASKOVIĆ
0036509921

Za proces Mrežu ću ubacivati ulomke koda, jer je predug za postaviti ga na jednu stranicu. Svi odsječci su iz do-while dijela koda.

```
if
:: (trenutni_predajnik == 0) ->
        if
        :: pred_ch2?oznaka_predajnik,oznaka_prijamnik,podaci,vrsta_poruke; izlaz = prij_ch2
        :: pred_ch1?oznaka_predajnik,oznaka_prijamnik,podaci,vrsta_poruke; izlaz = prij_ch1
        fi;
:: (trenutni_predajnik == 1) ->
        if
        :: prij_ch1? oznaka_predajnik,oznaka_prijamnik,podaci,vrsta_poruke; izlaz = pred_ch1
        :: pred_ch1? oznaka_predajnik,oznaka_prijamnik,podaci,vrsta_poruke; izlaz = prij_ch1
        fi;
:: (trenutni_predajnik == 2) ->
        if
        :: prij_ch2?oznaka_predajnik,oznaka_prijamnik,podaci,vrsta_poruke; izlaz = pred_ch2
        :: pred_ch2?oznaka_predajnik,oznaka_prijamnik,podaci,vrsta_poruke; izlaz = prij_ch2
        fi;
fi;
```

U ovom odsječku biramo kanal u odnosu na predajnik. Ako je vrijednost varijable trenutni_predajnik jednaka 0, to je početna situacija u kojoj postoji 50%-tna šansa da se odabere kanal koji povezuje prvi predajnik i mrežu te 50%-ta šansa da se odabere kanal koji povezuje drugi predajnik i mrežu.
Varijabla trenutni_predajnik mijenja vrijednost u odnosu na oznaku trenutnog predajnika.

```
if
        :: (trenutni_predajnik == 0) ->
                trenutni_predajnik = oznaka_predajnik;
                broj_aktivnih_predajnika++;
        :: else -> skip
fi;
```

U ovom odsječku provjeravamo koji predajnik je prvi poslao poruku te na osnovu toga povećava varijablu koja označava broj aktivnih predajnika za 1.

PERO PASKOVIĆ
0036509921

```
if
	:: (vrsta_poruke == 0) -> //ako je poruka paket, gubi se 1/3=33% poruka
		if
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,0;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,0;
		:: (podaci != 13) -> skip;
		fi;
	:: (vrsta_poruke == 1) -> //ako je poruka potvrda, gubi se 1/10=10% poruka
		if
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: izlaz!oznaka_predajnik,oznaka_prijamnik,podaci,1;
		:: (podaci != 13) -> skip;
		fi;
fi;
```

Ovaj odsječak koda simulira gubitak paketa koji se događa u 33%, tj 1/3 slučajeva te gubitak potvrde koji se događa u 10%, tj 1/10 slučajeva.

```
if
	:: (vrsta_poruke == 1 && broj_aktivnih_predajnika == 2 && podaci == 14) ->
		break;
	:: else -> skip
fi;
```

Ovaj odsječak osigurava izlazak iz do-while petlje ako je obrađeno svih 14 poruka te ako je primljena potvrda 14. poruke 2. pokrenutog predajnika. Varijabla broj_aktivnih_predajnika označava broj predajnika koji su dosad odašiljali poruke.

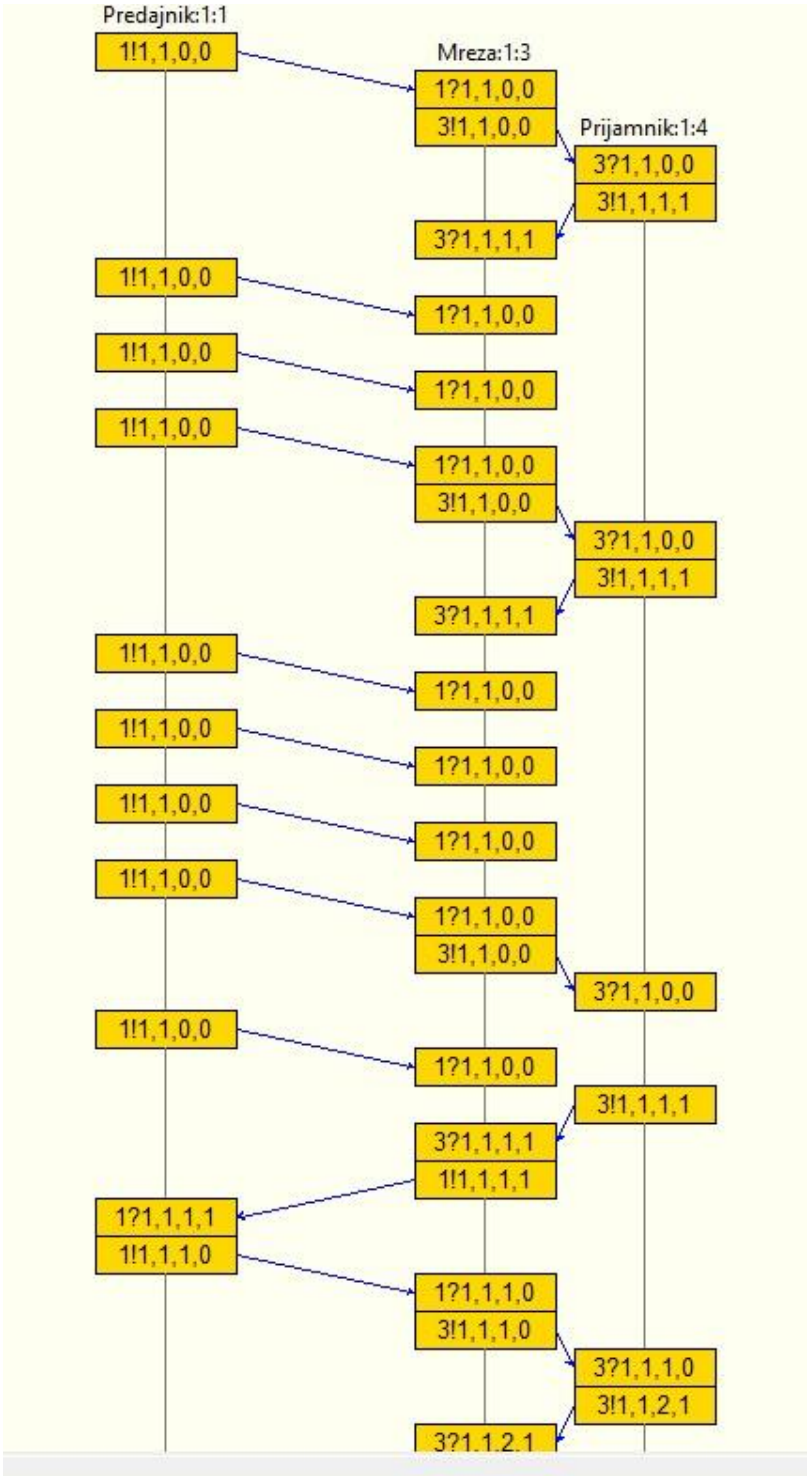
```
if
	:: (vrsta_poruke == 1 && podaci == 14) ->
		if
		:: (trenutni_predajnik == 1) -> trenutni_predajnik = 2; broj_aktivnih_predajnika++
		:: (trenutni_predajnik == 2) -> trenutni_predajnik = 1; broj_aktivnih_predajnika++
		fi;
	:: else -> skip
fi;
```

Ovaj odsječak simulira mijenjanje predajnika nakon što prvi predajnik po redu završi sa slanjem poruka primitkom potvrde 14. poruke.

PERO PASKOVIĆ
0036509921

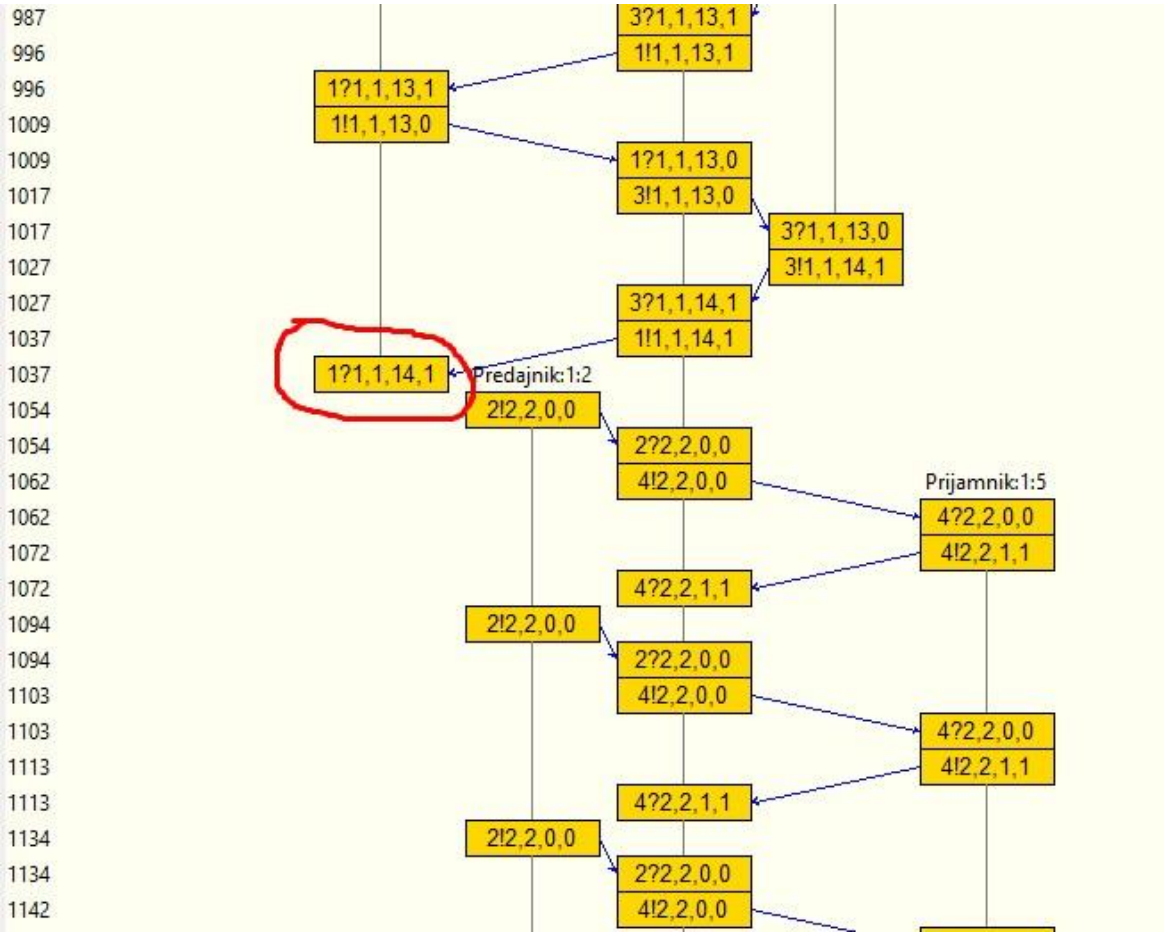
SIMULACIJA

Provedene su simulacije s više *seedova* (7, 123, 569, 850, 999) i svaka simulacija je uspješno došla do kraja.
Ovdje ću postaviti *screenshotove* simulacijse sa *seedom* 7.



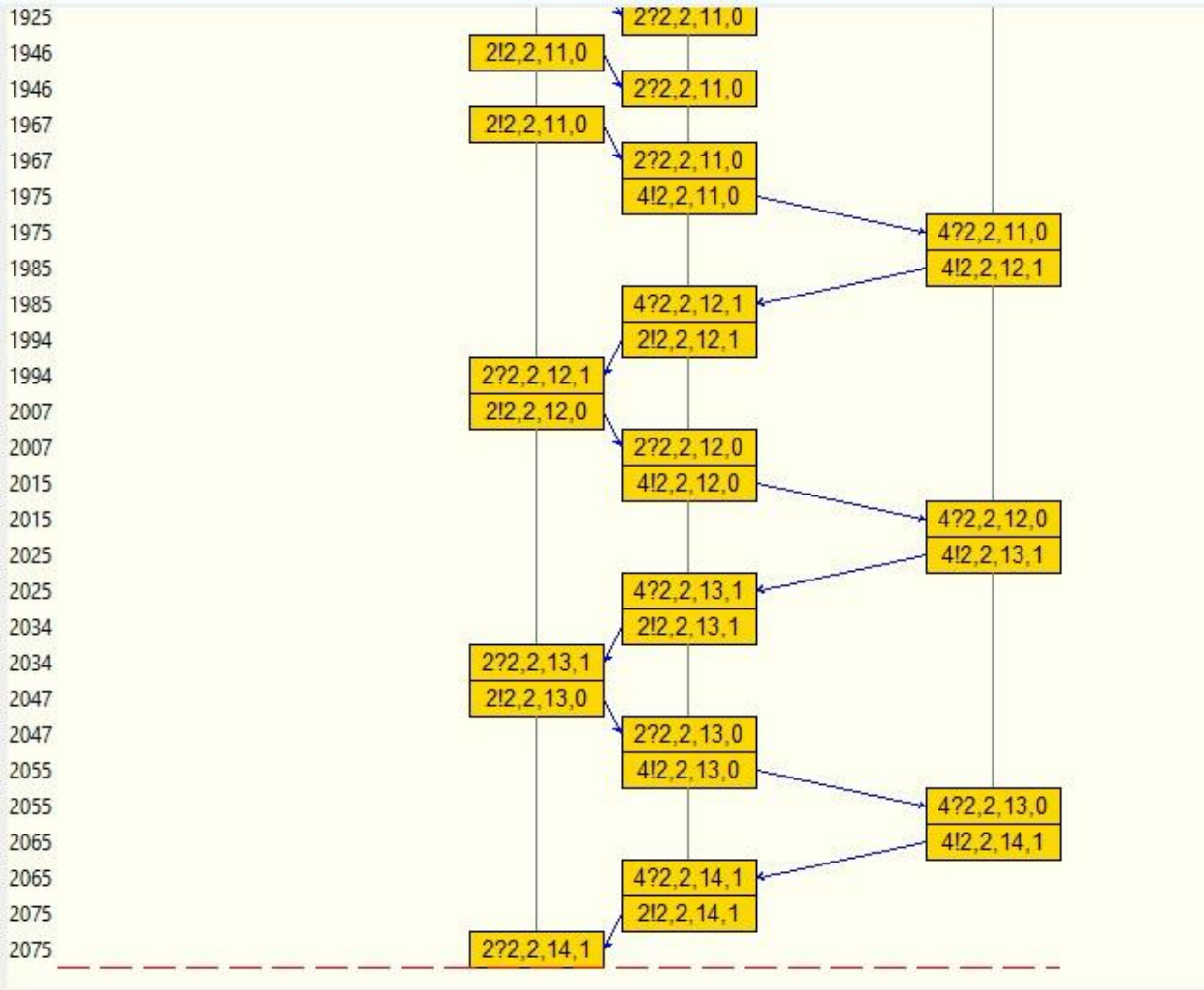
PERO PASKOVIĆ
0036509921

Na ovom *screenshotu* simulacije vidimo početak iste te odmah možemo primijetiti brojne retransmisije u slučaju isteka vremena (timeout). Ovdje su aktivni kanali pred_ch1 (oznaka 1) te prij_ch1 (oznaka 3). Na samom početku možemo vidjeti i gubitak potvrde.



Na ovom *screenshotu* možemo vidjeti promjenu predajnika nakon primitka potvrde 14. poruke prvog predajnika (crvena oznaka) te početak izmjene poruka između predajnika 2 i prijamnika 2 putekm kanala pred_ch2 (oznaka 2) i prij_ch2 (oznaka 4).

PERO PASKOVIĆ
0036509921



Na ovom *screenshotu* možemo vidjeti kraj razašiljanja poruka između predajnika 2 i prijamnika 2 dobitkom potvrde 14. poruke što je ujedno i kraj simulacije.

PERO PASKOVIĆ
0036509921

VERIFIKACIJA

```
spin -a ver2.pml
gcc -DMEMLIM=1024 -O2 -DXUSAFE -DSAFETY -DNOCLAIM -w -o pan pan.c
./pan -m10000  -c0
Pid: 31224
pan:1: invalid end state (at depth 849)

(Spin Version 6.5.1 -- 20 December 2019)
        + Partial Order Reduction

Full statespace search for:
        never claim         - (not selected)
        assertion violations +
        cycle checks        - (disabled by -DSAFETY)
        invalid end states +

State-vector 216 byte, depth reached 877, errors: 9
    9097 states, stored
    1843 states, matched
   10940 transitions (= stored+matched)
       0 atomic steps
hash conflicts:         0 (resolved)

Stats on memory usage (in Megabytes):
   1.978   equivalent memory usage for states (stored*(State-vector + overhead))
   1.356   actual memory usage for states (compression: 68.53%)
           state-vector as stored = 144 byte + 12 byte overhead
  64.000   memory used for hash table (-w24)
   0.343   memory used for DFS stack (-m10000)
  65.613   total actual memory usage


unreached in init
        (0 of 6 states)
unreached in proctype Predajnik
        (0 of 14 states)
unreached in proctype Mreza
        (0 of 76 states)
unreached in proctype Prijamnik
        (0 of 10 states)

pan: elapsed time 0.012 seconds
To replay the error-trail, goto Simulate/Replay and select "Run"
```

Ovdje možemo vidjeti prikaz verifikacije simulacije sa *seedom* 7. Ovakav prikaz je i za sve ostale *seedove*.
Na *screenshotu* se jasno vidi da nema nedohvatljivih stanja te da nema grešaka.