30 Апреля 2010 (?)

Инсерционное моделирование 1

Лекция 12

Примеры инсерционных машин

Императивные программы (sequential_imperative)

```
ins:=rs(E,H,P,Q,a,x,y,u)(
    E[0]                    = 0,
    E[bot]                  = bot,
    E[P+Q]                  = E[P]+E[Q],
    E[insert.P]             = insert_proc(E,P),
    E[define_env H.P]       = H[P],
    E[(x:=y).P]             = assign_proc(E,x,y,P),
    E[check(u,x,y).P]       = if(
        compute_obj(E,u),
        E[x;P],
        E[y;P]
    ),
    E[a.P]                  = a.E[P],
    E[P]                    = E[short_intens P]
);
```

```
(
    define_env obj(
        i:Nil,
        x:50,
        y:Nil,
        fact:Nil
    );
    y:=1;
    for(i:=1,i<=x,i:=i+1,
        y:=y*i
    );
    fact:=y
);
```

```
unfold_rs:=rs(m,x,y,z,u,P,Q)(
    (x; y) = seq(x;y),
/* Call external environment */
    doprog P  = Fdo P & Delta,
    wait = doprog (wait_ent()),
/* insertion */
    make_insertion = Mesg("\ninsertion").insert,
/* imperative */
    (u->P else Q) = check(u,P,Q),
    while(u,   P) = check(u,(P;while(u,P)),Delta),
    for(x,y,z, P) = (x;while(y,(P;z)))
);
```

```
assign_proc:=proc(E,x,y,P)(
    E.x-->compute_obj(E,y);
    return E[P]
);
```

Трассы

```
ins:=rs(E,H,P,Q,a,x,y,u)(
   E[0]                    = 0,
   E[bot]                  = bot,
   E[P+Q]                  = E[P]+E[Q],
   E[insert.P]             = insert_proc(E,P),

   E[print_val x.P]        = put("\n"x"="(E.x))&E[P],
   E[define_env H.P]       = define_env.H[P],
   E[(x:=y).P]             = (x:=y).assign_proc(E,x,y,P),
   E[check(u,x,y).P]       = u.if(
      compute_obj(E,u),
      E[x;P],
      E[y;P]
   ),

   E[a.P]                  = a.E[P],
   E[P]                    = E[short_intens P]
);
```

Строгое параллельное погружение

```
ins:=rs(x,m,u,v,a,c,p,E,F,G,H,P,Q,R)(
    Delta[0]                = 0,
    Delta[bot]              = bot,
    Delta[Delta]            = Delta,
    Delta[P+Q]              = Delta[P]+Delta[Q],
    Delta[insert.P]         = insert_proc(P),
    Delta[a.P]              = a.Delta[P],
    Delta[P]                = Delta[short_intens P],
    E[P]                    = Delta[E||P]
);
```

```
(((a;b)||(a;b));a+b);

(a||b)||(a||(b;0));

(
    (example 1:(a||b))+
    (example 2:(a||b||c))+
    (example 3:(a||((b||c);0))+d)+
    (example 4:((a;b)||(a;b)));
    make_insertion
);
```

```
unfold_rs:=rs(n,x,y,P)(
/* General compositions */
    (x; y) = seq(x;y),
     x||y  = intens(synchr(x,y)+lmrg(x,y)+lmrg(y,x)),
     x|^1  = x,
     x|^2  = intens(synchr(x,x)+lmrg(x,x)),
     x|^n  = x||(x|^(n-1)),
/* Reading AL program */
……
/* Call external environment */
……
/* insertion */
……
);
```

```
combine:=rs(x,y)(
    x >< y = mrg(x >< y)
)
```

Достижимость

```
ins:=rs(x,m,u,v,a,c,p,E,F,G,H,P,Q,R)(
    Delta[0]                = 0,
    Delta[bot]              = bot,
    Delta[Delta]            = Delta,
    Delta[P+Q]              = Delta[P]+Delta[Q],
    Delta[insert.P]         = insert_proc(P),
    Delta[a.P]              = a.Delta[P],
    Delta[P]                = Delta[short_intens P],
    E[P]                    = Delta[E||P]
);
```

```
find_termination(

    (A:x to B).(A:y from B)||
    (B:x from A).(B:y to A)

);

find_termination(a||(a;b));
find_termination((a;b)||(a;b));
find_dead_lock((a||b)||(a||(b;0)));
find_dead_lock((a||b)||(a+b;0));
```

```
unfold_rs:=rs(n,x,y,P)(
/* General compositions */
    (x; y) = seq(x;y),
     x||y  = intens(synchr(x,y)+lmrg(x,y)+lmrg(y,x)),
     x|^1  = x,
     x|^2  = intens(synchr(x,x)+lmrg(x,x)),
     x|^n  = x||(x|^(n-1)),
/* Reading AL program */
……
/* Call external environment */
……
/* insertion */
……
);
```

```
combine:=rs(x,y)(
    x >< y = mrg(x >< y)
)
```