Esterel de A à Z

5. Boucles et réincarnation en Esterel

Gérard Berry

Collège de France
Chaire Algorithmes, machines et langages

Cours du 14 mars 2018

Suivi du séminaire de Reinhard von Hanxleden

COLLÈGE DE FRANCE
1530

gerard.berry@college-de-france.fr
http://www-sop.inria.fr/members/Gerard.Berry

Agenda

1. Pourquoi une étude spéciale des boucles

Pourquoi une étude spéciale des boucles

- Les boucles ne posent pas de problème sémantique

$$\frac{\overline{p} \xrightarrow[I]{O,\,k} \overline{p}' \qquad k>0}{\texttt{loop}\ \overline{p}\ \texttt{end} \xrightarrow[I]{O,\,k} \texttt{loop}\ \overline{p}'\ \texttt{end}} \qquad (go\text{-}res\text{-}loop\text{-}k{>}0)$$

$$\frac{\hat{p} \xrightarrow[I]{O,\,0} p \qquad p \xrightarrow[I]{O',\,k} \overline{p}' \qquad k>0}{\texttt{loop}\ \hat{p}\ \texttt{end} \xrightarrow[I]{O\cup O',\,k} \texttt{loop}\ \overline{p}'\ \texttt{end}} \qquad (res\text{-}loop\text{-}0)$$

- Interprète Esterel trivial : implémenter les règles

Pourquoi une étude spéciale des boucles

- Mais la traduction en circuits du cours 4 ne marche pas car pas d'allocation dynamique de portes et fils
- Aussi nécessaire pour les implémentations en logiciel rapide (S. Edwards, D. Potop, M. Perreaut), cf. cours 6

But de ce cours : traiter la schizophrénie des signaux et parallèles par la réincarnation

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène

Exemple de programme schizophrène

Un programme banal :

```
output O ;
loop
   signal S in
      if S then emit O end ;
      pause ;
      emit S
   end signal
end loop
```

Instant 1

Un programme banal :

```
→ output O ;
  loop
→   signal S in
      if S then emit O end ;
→     pause ;
      emit S
    end signal
  end loop
```

Arrêt sur pause, S absent, O non émis,

Instant 2a

Un programme banal :

```
output O ;
loop                                          S
   signal S in
      if S then emit O end ;
      pause ;
      emit S
   end signal
end loop
```

S,O inconnus, pause termine, la séquence démarre

Instant 2b

Un programme banal :

```
output O ;
loop                                          S
   signal S in                                S
      if S then emit O end ;
      pause ;
      emit S
   end signal
end loop
```

emit S est exécuté, S est marqué présent

Instant 2c

Un programme banal :

```
output O ;
loop
   signal S in
      if S then emit O end ;
      pause ;
      emit S
   end signal
end loop
```

S n'a plus d'existence

S
S
S

L'instruction signal se termine, on sort de sa portée

Instant 2d

Un programme banal :

```
output O ;
loop                                          S
   signal S in                                S
      if S then emit O end ;                  S
      pause ;
      emit S
   end signal
end loop
```

La boucle se termine

Instant 2e

Un programme banal :

```
output O ;
loop                                          S
   signal S in                                S
      if S then emit O end ;                  S
      pause ;
      emit S
   end signal
end loop
```

La boucle redémarre

Instant 2f

Un programme banal :

```
output O ;
loop                                          S
   signal S in                                S
      if S then emit O end ;                  S
      pause ;                                 S
      emit S
   end signal
end loop
```

S est redéclaré et remarqué inconnu (S)

Instant 2g

Un programme banal :

```
output O ;
loop                                          S
   signal S in                                S
      if S then emit O end ;                  S
      pause ;                                 S
      emit S                                  S
   end signal
end loop
```

N’ayant pas d’émetteurs accessibles, S est absent

Instant 2h

Un programme banal :

```
output O ;
loop                                          S
   signal S in                                S
      if S then emit O end ;                  S
      pause ;                                 S
      emit S                                  S
   end signal
end loop
```

Le test se termine sans émettre O, qui est donc absent

Instant 2i

Un programme banal :

```
output O ;
loop                                          S
   signal S in                                S
      if S then emit O end ;                  S
   ⟶  pause ;                                 S
      emit S                                  S
   end signal
end loop
```

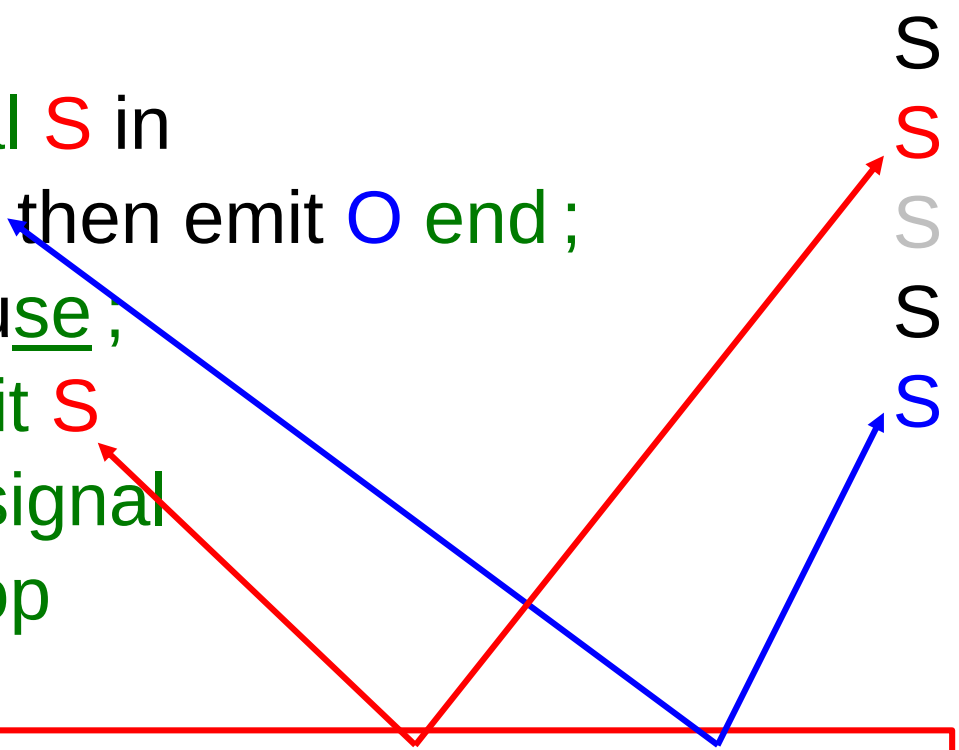
Le pause s'allume, fin de l'instant

S est schizophène !

Un programme banal :

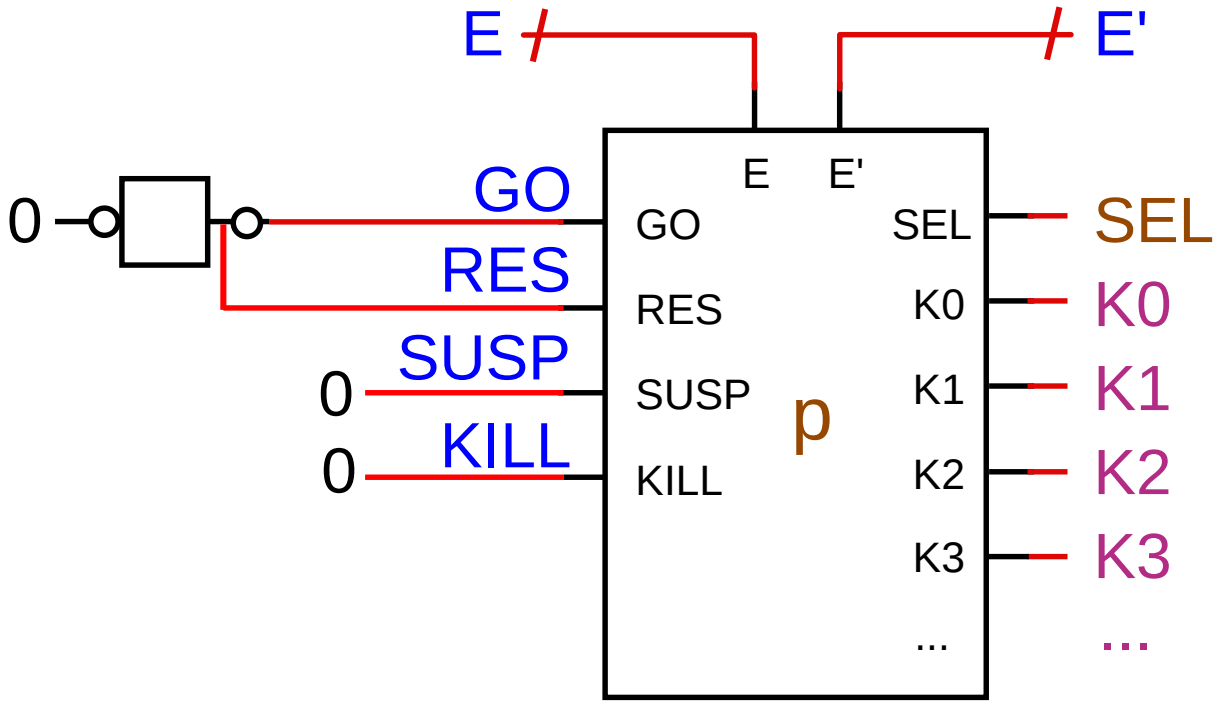
```
output O ;
loop
   signal S in
      if S then emit O end ;
      pause ;
      emit S
   end signal
end loop
```

S
S
S
S
S



S est simultanément présent et absent !

Oubli du cours 4 : boot d'un circuit



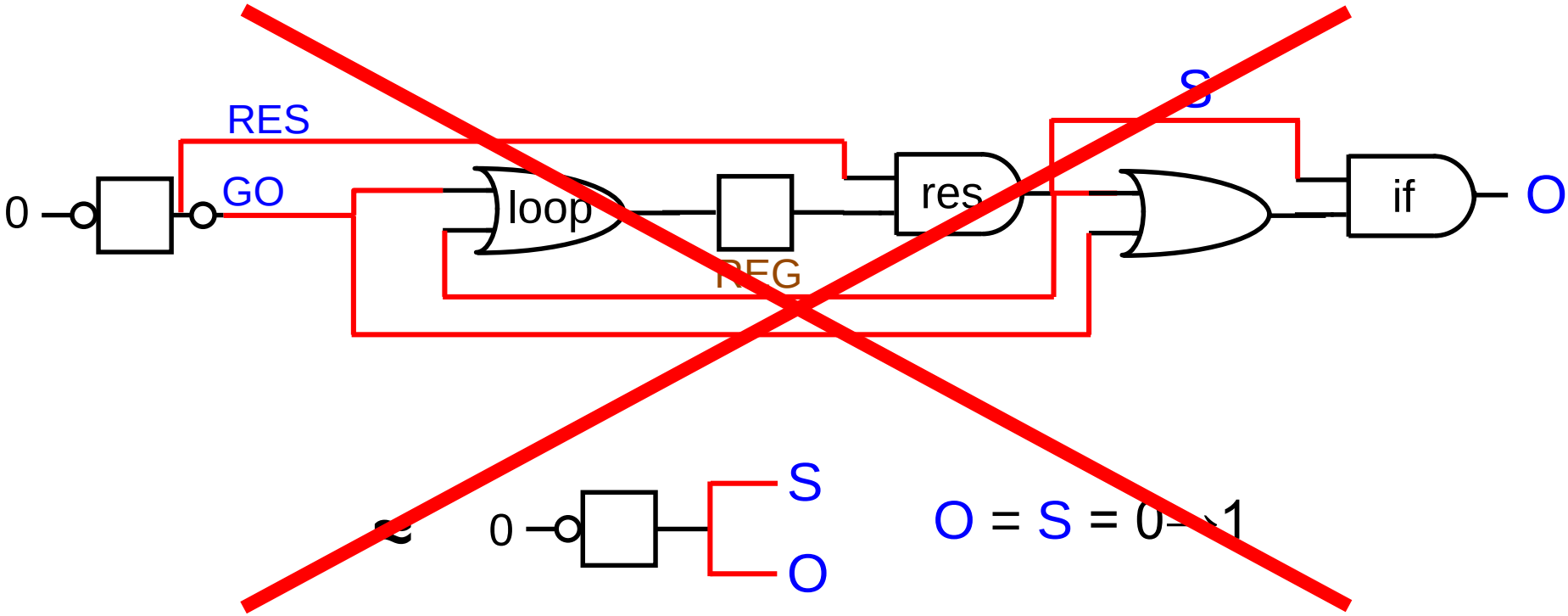
GO = 1→0

RES = 0→1

SUSP = 0

KILL = 0

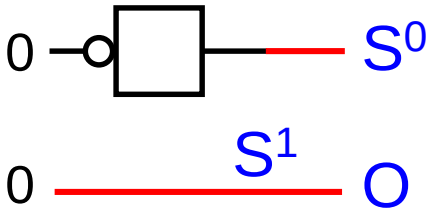
Circuit trop simple du programme précédent



≠

$O = S = 0 \rightarrow 1$

≠



$S^0 = 0 \rightarrow 1$

$O = S = 0$

Les parallèles aussi sont schizophrènes

```
loop
   { pause || pause };
   emit O
end loop
```

programme source

```
loop
   { pause || pause };
   emit O
end loop
```

instant 1

```
loop
   { pause || pause };
   emit O
end loop
```

instant 2a

synchro → 0

(terminaison)

```
loop
   { pause || pause };
   emit O
end loop
```

instant 2b

synchro → 1

(pause)

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène

Exemple de double schizophrénie

```
loop
  signal S1 in
    trap T in
      loop
        signal S2 in
          if S1 then emit S2 end ;
          pause ;
        end signal
      end loop
    ||
      pause ;
      emit S1 ;
      exit T
    end trap
  end signal
end loop
```

Instant 1a : S1, S2 alloués mais inconnus

S1 S2

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
? ⟶          if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
⟶    ||
⟶       pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

Instant 1b : S1, S2 absents car non émissibles

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2

S1 S2

Instant 1c : fin de la réaction

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2

S1 S2

Instant 2a : reprise, S1 et S2 inconnus

S1 S2

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

Instant 2b : S1 présent car émis

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2

S1

Instant 2c : S2 absent, car non émissible

```
loop
   signal S1 in
      trap T in
         loop
         signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2
S1
S2

pas d'émetteur accessible
d'ici jusqu'au end signal
qui signe la fin de vie de ce S2

Instant 2d : S2 plus visible

```
loop
   signal S1 in
      trap T in
         loop
   ——→      signal S2 in
               if S1 then emit S2 end ;
               pause ;
   ——→      end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2

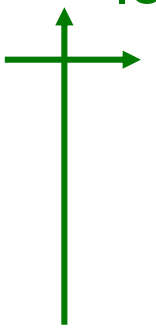
S1

S2

S2

Instant 2e : rebouclage et réallocation de S2

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```



```
S1 S2
S1
     S2
     S2
     S2
```

Instant 2f : S2 présent car émis

```
loop
   signal S1 in
      trap T in
         loop
      ⟶   signal S2 in
        ⟶    if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2

S1

S2

S2

S2

S2

Instant 2g : pause

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1	S2
S1	
	S2
	S2
	S2
	S2

Instant 2h : sortie de trappe, S2 plus visible

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2

S1

S2

S2

S2

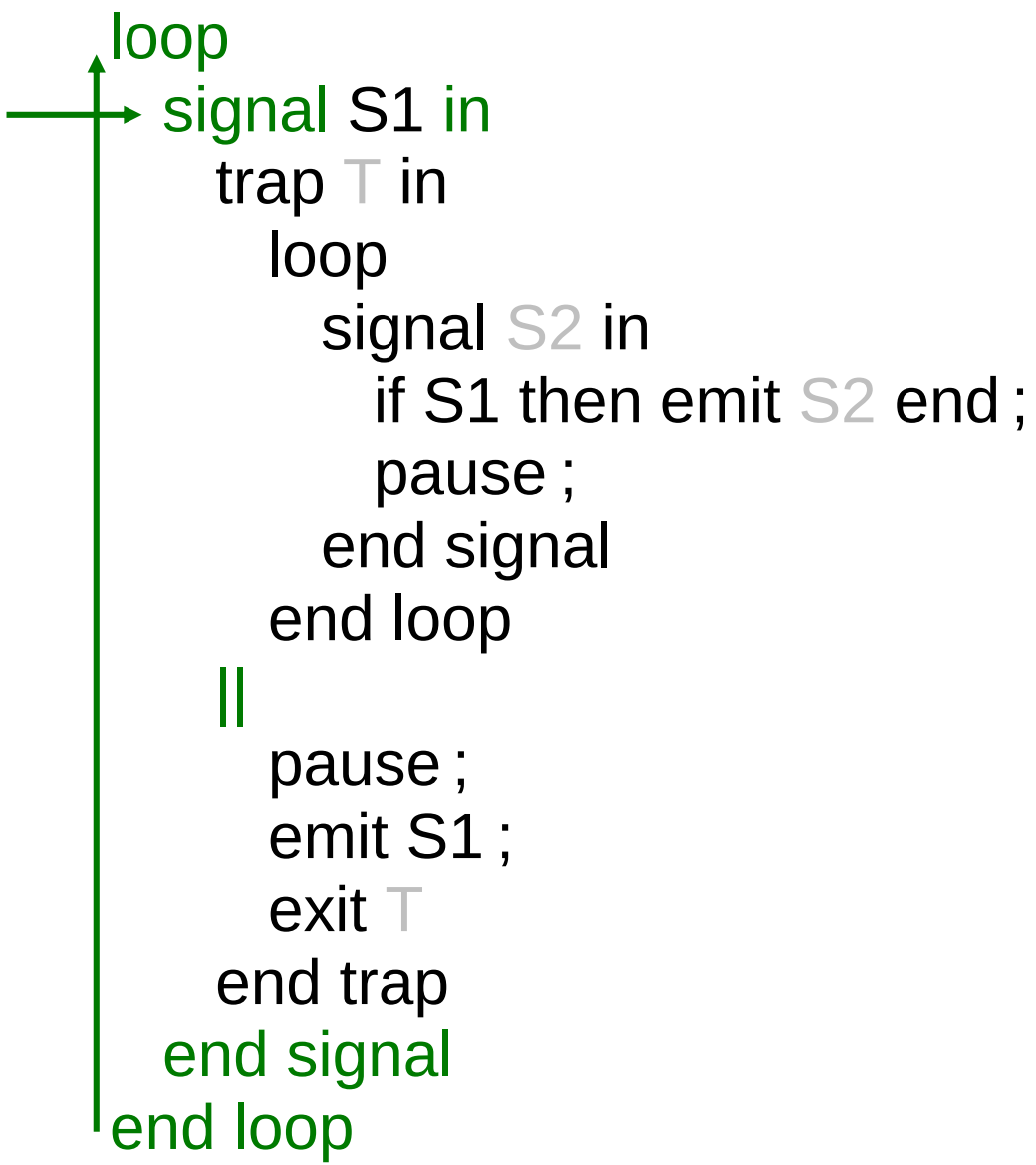
S2

Instant 2i : S1 plus visible

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

S1 S2
S1
S2
S2
S2
S2
S2
S1

Instant 2j : rebouclage, S1 inconnu



S1 S2
S1
S2
S2
S2
S2
S2
S1
S1

Instant 2k : S1 absent car non émissible

```
loop
  signal S1 in
    trap T in
      loop
        signal S2 in
          if S1 then emit S2 end ;
          pause ;
        end signal
      end loop
    ||
      pause ;
      emit S1 ;
      exit T
    end trap
  end signal
end loop
```

S1 S2
S1
S2
S2
S2
S2
S1
S1
S1

Instant 2l : S2 inconnu

```
loop
   signal S1 in
    trap T in
       loop
    ⟶  signal S2 in
          if S1 then emit S2 end ;
          pause ;
        end signal
       end loop
    ||
       pause ;
       emit S1 ;
       exit T
    end trap
   end signal
end loop
```

S1	S2
S1	
	S2
	S2
	S2
	S2
	S2
S1	
S1	
S1	
	S2

Instant 2m : S2 absent car non émissible

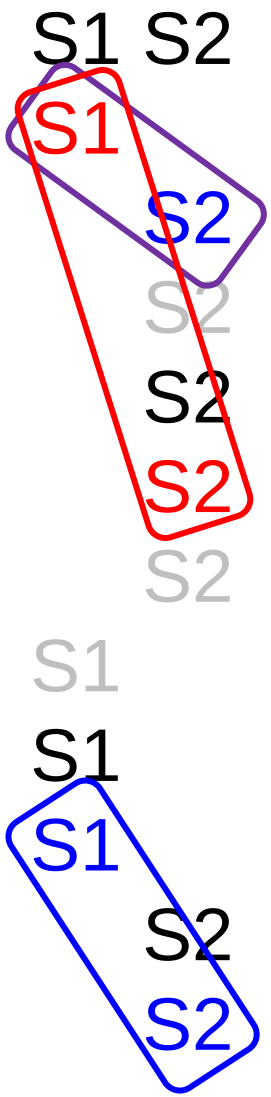
```
loop
  signal S1 in
    trap T in
      loop
        signal S2 in
          if S1 then emit S2 end ;
          pause ;
        end signal
      end loop
    ||
      pause ;
      emit S1 ;
      exit T
    end trap
  end signal
end loop
```

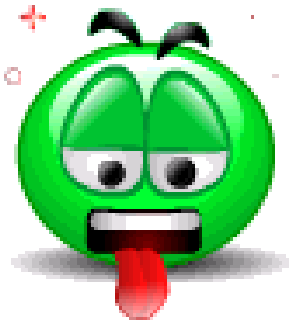
S1	S2
S1	
	S2
	S2
	S2
	S2
	S2
S1	
S1	
S1	
	S2
	S2

Instant 2o: c'est fini !

```
loop
   signal S1 in
      trap T in
         loop
            signal S2 in
               if S1 then emit S2 end ;
               pause ;
            end signal
         end loop
      ||
         pause ;
         emit S1 ;
         exit T
      end trap
   end signal
end loop
```

Deux S1, trois S2
trois combinaisons,
S2 doublement schizo !





D’accord, c’est du boulot, mais du boulot d’ordinateur !

La bonne nouvelle :
tout cela reste parfaitement sain, car les réallocations se suivent strictement sans jamais interférer entre elles !

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène
4. Traiter la schizophrénie par duplication du corps

Traiter la schizophrénie par duplication du corps

loop p end ≈ loop p ; p end

```
loop
|  signal S in
|     if S then emit O end
|     pause ;
|     emit S
|  end signal ;
end loop
```

≈

```
loop
|  signal S in
|     if S then emit O end
|     pause ;
|     emit S
|  end signal ;
|  signal S in
|     if S then emit O end
|     pause ;
|     emit S
|  end signal
end loop
```

Instant 1 : S^0, O absents car non émissible

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

Instant 2a : S^0 présent car émis

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

Instant 2a : S^0 présent car émis

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

Instant 2b : S^0 plus visible, S^1 inconnu

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

Instant 2b : S^1, O absents, fin de l'instant

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

Instant 3a : S^1 présent car émis

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

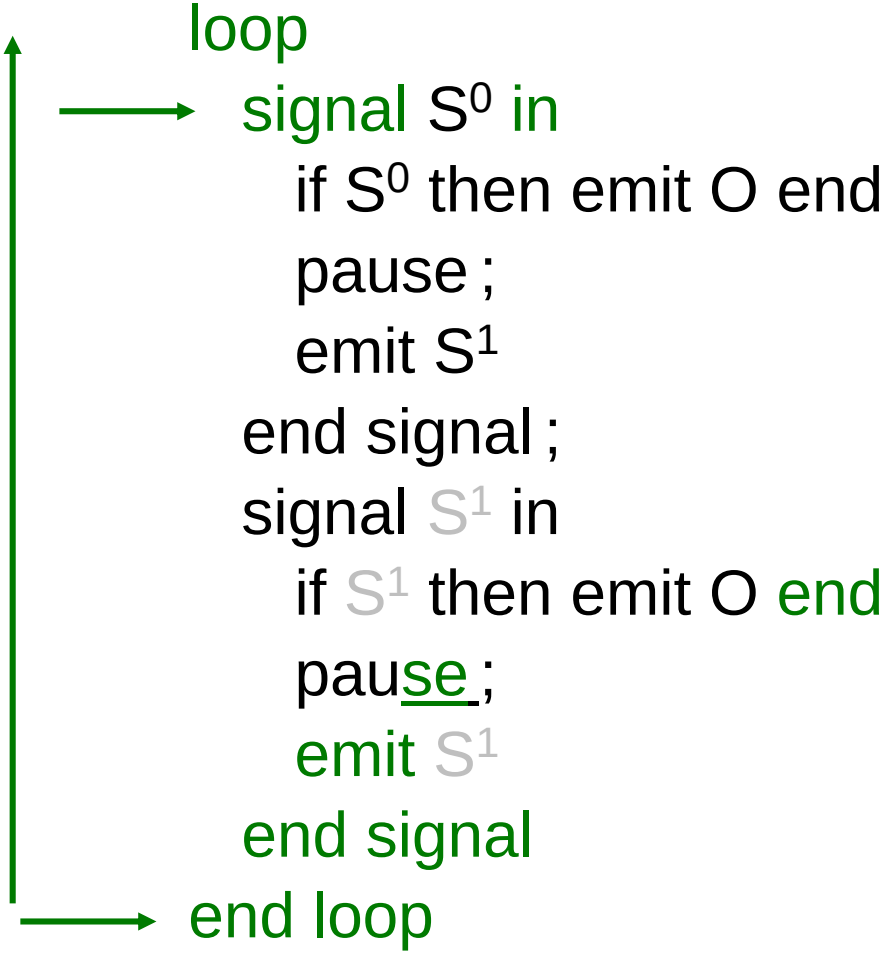
Instant 3a : S^1 présent car émis

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

Instant 3b : S^1 invisible, boucle, S^0 inconnu

loop p end ≈ loop p ; p end



Instant 3b : S^0, O absents, fin de l'instant

loop p end ≈ loop p ; p end

```
loop
   signal S0 in
      if S0 then emit O end
      pause ;
      emit S0
   end signal ;
   signal S1 in
      if S1 then emit O end
      pause ;
      emit S1
   end signal
end loop
```

Résout le problème, en théorie

- Solution simple, qui marche aussi pour le parallèle

Mais ne fonctionne pas en pratique :

imbrication de boucles, signaux et parallèles

$\Longrightarrow$ explosion exponentielle du code !

Mais danger : explosion exponentielle !

```
loop
   signal S1 in
      loop
         signal S2 in
            loop
               signal S3 in
               ...
               end signal
            end loop
         end signal
   end signal
end loop
```

```
loop
   signal S1 in
      loop
         signal S2 in
            loop
               signal S3 in
               ...
               end signal ;
               signal S3 in
               ...
               end signal ;
            end loop
         end signal
   end signal
end loop
```

Mais danger : explosion exponentielle !

```
loop
   signal S1 in
      loop
         signal S2 in
            loop
               signal S3 in
               ...
               end signal
            end loop
         end signal
   end signal
end loop
```

```
loop
  signal S1 in
    loop
      signal S2 in
        loop
          signal S3 in
          ...
          end signal ;
          signal S3 in
          ...
          end signal ;
        end loop
      end signal ;
      signal S2 in
        loop
          signal S3 in
          ...
          end signal ;
          signal S3 in
          ...
          end signal ;
        end loop
      end signal
  end signal
end loop
```

Mais danger : explosion exponentielle !

```
loop
   signal S1 in
      loop
         signal S2 in
            loop
               signal S3 in
               ...
               end signal
            end loop
         end signal
   end signal
end loop
```

```
loop
  signal S1 in
    loop
      signal S2 in
        loop
          signal S3 in
          ...
          end signal ;
          signal S3 in
          ...
          end signal ;
        end loop
      end signal;
      signal S2 in
        loop
          signal S3 in
          ...
          end signal ;
          signal S3 in
          ...
          end signal ;
        end loop
      end signal
  end signal
  signal S1 in
    loop
      signal S2 in
        loop
          signal S3 in
          ...
          end signal ;
          signal S3 in
          ...
          end signal ;
        end loop
      end signal;
      signal S2 in
        loop
          signal S3 in
          ...
          end signal ;
          signal S3 in
          ...
          end signal ;
        end loop
      end signal
  end signal
end loop
```

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène
4. Traiter la schizophrénie par duplication du corps
5. Plus efficace : réincarner la surface des boucles

Un traitement efficace

Pour traiter de dédoublement de la personnalité
des signaux et parallèles (schizophrénie),
comprendre la théorie et la pratique
de leurs réincarnations !

Instant 2 : les deux incarnations

Un programme banal :

```
output O ;
loop
   signal S in
      if S then emit O end ;
      pause ;
      emit S
   end signal
end loop
```

Matsya

Kurma

Surface et profondeur d'une instruction

```
output O ;
loop
   signal S in
      if S then emit O end ;
      pause ;
      emit S
   end signal
end loop
```

surface : ce qui est accessible instantanément à partir du début de l'instruction

profondeur : le reste, là où sont les pauses des p̂

Surface instantanée, profondeur longue

```
signal S in
    if S then emit O end ;
    emit X
    pause ;
    emit S ;
    pause ;
    pause ;
    emit S
end signal
```

surface et profondeur peuvent s'interpénétrer

```
signal S in
    if S then
      emit O
    else
      pause
    end ;
    emit X
    pause ;
    emit S ;
    pause ;
    pause ;
    emit S
end signal
```

Pas de souci particulier

zone mixte :
accessible soit du emit O,
soit du pause

Tardieu-v1 : supprimer la boucle

```
output O ;
signal S in
      if S then emit O end ;
      1 : pause ;
      emit S
end signal ;
signal S in
      if S then emit O end ;
      gotopause 1;
 end signal ;
```

Evite la duplication de la profondeur

Tardieu-v1 : supprimer la boucle

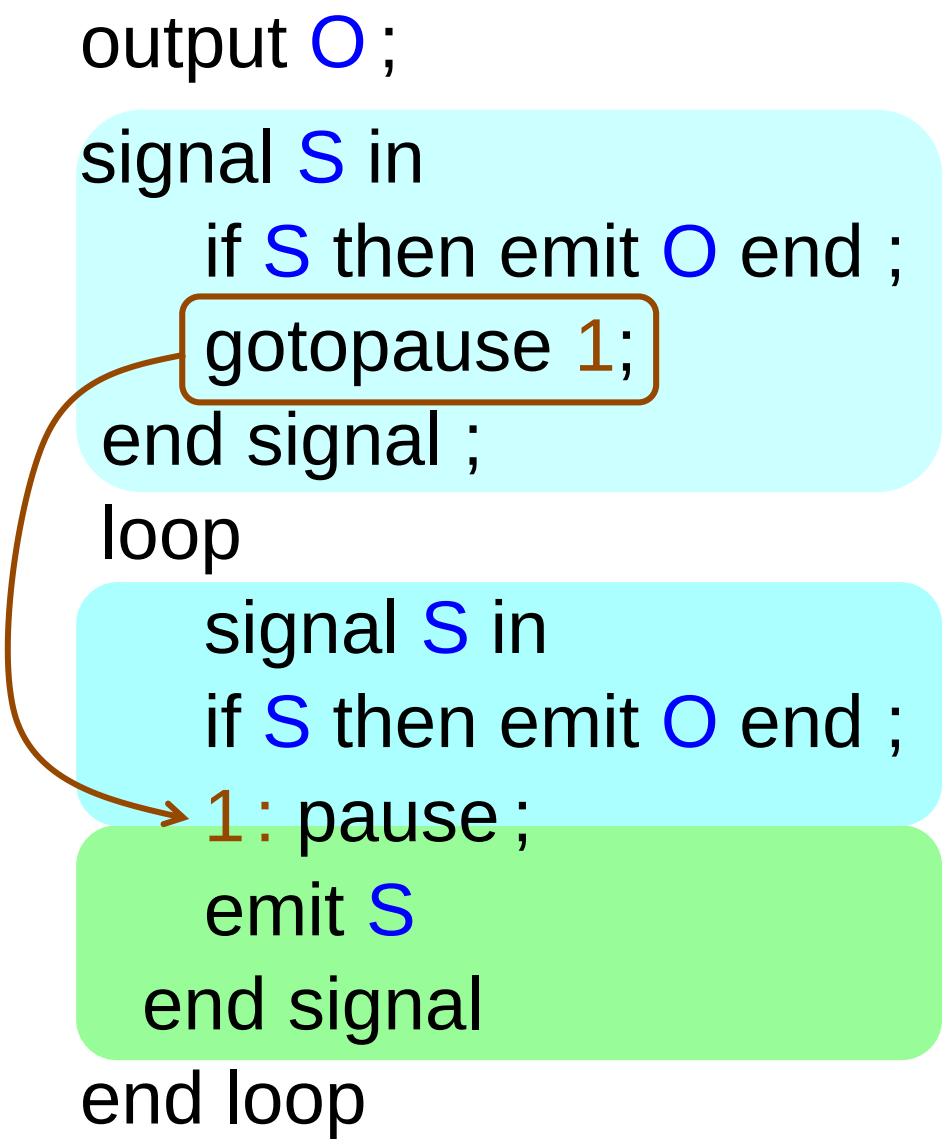
```
output O ;
signal S in
    if S then emit O end ;
    1 : pause ;
    emit S
end signal ;
signal S in
    if S then emit O end ;
    gotopause 1;
end signal ;
```

Matsya

Kurma

Evite la duplication de la profondeur

Tardieu-v2 : dupliquer la surface en tête



Tardieu-v2 : dupliquer la surface en tête

```
output O ;
signal S in
      if S then emit O end ;
      gotopause 1;
 end signal ;
 loop
      signal S in
      if S then emit O end ;
      1 : pause ;
      emit S
   end signal
end loop
```

Matsya

Kurma

Répliquer la surface → expansion quadratique

```
loop
  signal S1 in
    loop
      signal S2 in
        loop
          signal S3 in
            surface
            profondeur
          end signal
        end loop
    end loop
  end signal
end loop
```

```
signal S1 in
  signal S2 in
    signal S3 in
      surface [g/p]
    end signal
  end signal
end signal ;
```

```
loop signal S1 in
  signal S2 in
    signal S3 in
      surface [g/p]
    end signal
  end signal ;
  loop signal S2 in
    signal S3 in
      surface [g/p]
    end ;
    loop signal S3 in
      surface ;
      profondeur
    end end
  end end
end end
```

$$1\times S1 + 2\times S2 + 3\times S3 + \ldots + n\times Sn \approx n^2$$

Tardieu : les résultats

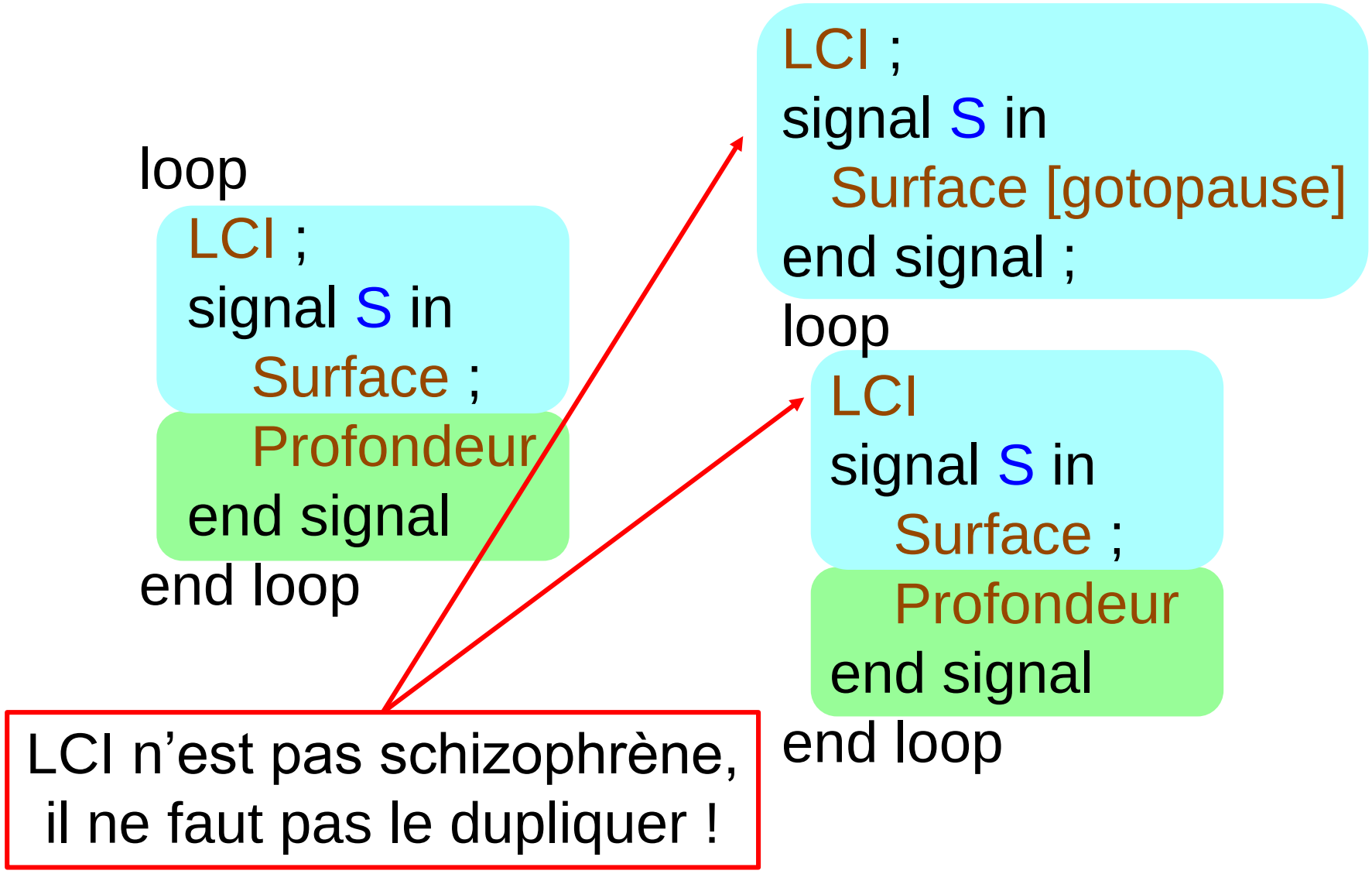
- v1, supprimer la boucle : simple
- v2 : réincarner en tête : mieux optimisable
- Les deux méthodes sont quadratiques 😁
- Inconvénient : modifient et allongent le code source (debug…)

Thèse d'Olivier Tardieu :
présentation algébrique élégante
+ optimisations subtiles et élégantes

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène
4. Traiter la schizophrénie par duplication du corps
5. Plus efficace : réincarner la surface des boucles
6. Indexer les incarnations

Ne pas dupliquer le LCI (Long Code Instantané)



GB : étiqueter le source

```
        output O ;
pmax    loop
 /0         LCI1 ;
 /1         signal S in
      0/1     if S then emit O end ;
              pause ;
      1/1     emit S
 /0         end signal ;
            LCI2
        end loop
```

surface = 0/1
profondeur =1/1

La schizophrénie n'est pas due à la boucle seule, mais à l'interaction loop / signal (ou loop / ||) !

Les index d'incarnations

- Paire d'entiers l/d avec l ≤ d
- l : niveau d'incarnation (*level*)
- d : niveau de profondeur (*depth*)
- l = 0..d−1 : surfaces de plus en plus profondes
- l = d : profondeur
- push(l/d) = l/(d+1)

  exemples : push(1/3) = 1/4, push(3/3) = 3/4
- pop(l/d) = min(l,d−1)/d−1

  exemples : pop(1/3) = 1/2, pop(2/3) = pop(3/3) = 2/2
- prof(l/d) = d/d

GB : étiquetage de la surface

pmax /0

output O$^{0/0}$;
loop$^{0/0}$
  LCI1$^{0/0}$;$^{0/0}$
  signal$^{0/0}$ S$^{0/1}$ in
/1 if$^{0/1}$ S$^{0/1}$ then emit$^{0/1}$ O$^{0/0}$ end$^{0/1}$;$^{0/1}$
    $^{0/1}$pause ;
    emit S
/0 end signal ;
  LCI2
end loop

push(0/0)

Matsya

GB : étiquettes de reprise → prof(0/1) = 1/1

pmax /0

output O$^{0/0}$;
loop$^{0/0}$
  LCI1$^{0/0}$; $^{0/0}$
  signal$^{0/0}$ S$^{0/1,1/1}$ in
    if$^{0/1}$ S$^{0/1}$ then emit$^{0/1}$ O$^{0/1}$ end$^{0/1}$;$^{0/1}$
    $^{0/1}$pause$^{1/1}$;$^{1/1}$
    emit S$^{1/1}$
/0 end signal$^{0/0}$;$^{0/0}$
  LCI2$^{0/0}$
end loop$^{0/0}$

prof(0,1)

pop(1,1)

Kurma

Pas de duplication des LCI

GB : cacher la profondeur dans pmax

Matsya

Kurma

pmax /0

output O^0 ;
loop^0
  $LCI1^0$; ^{0}
  signal^0 $S^{0,1}$ in
    if^0 S^0 then emit^0 O^0 end^0 ;^{0}
    ^{0}pause^1 ;^{1}
    emit S^1
  end signal^0 ;^{0}
  $LCI2^0$
end loop^0

/1

/0

Pas de duplication des LCI

Instant1, 0/0

```
pmax   output O⁰ ;                                   S⁰ S¹
 /0    loop⁰
         LCI1⁰ ;⁰
         signal⁰ S⁰,¹ in
 /1        if⁰ S⁰ then emit⁰ O⁰ end ;⁰
           ⁰pause¹ ;¹
           emit¹ S¹
 /0      end signal⁰ ;⁰
         LCI2⁰
       end loop⁰
```

Instant 2a, reprise avec 1/1

pmax

/0

/1

/0

output O^0 ;

loop^0

  LCI1^0 ;^{0}

  signal^0 $S^{0,1}$ in

    if^0 S^0 then emit^0 O^0 end ;^{0}

    ^{0}pause^1 ;^{1}

    emit^1 S^1

  end signal^0 ;^{0}

  LCI2^0

end loop^0

S^0 S^1

S^1

S^1 présent

Instant 2b :sortie du signal, pop(1/1)=0/0

pmax /0 /1 /0

```
output O0 ;                                   S0 S1
loop0                                            S1
   LCI10 ;0                                   S0
   signal0 S0,1 in
      if0 S0 then emit0 O0 end ;0
      0pause1 ;1
      emit1 S1
   end signal0 ;0
   LCI20
end loop0
```

Rebouclage, 0/0

```
pmax   output O⁰ ;
 /0    loop⁰
          LCI1⁰ ;⁰
          signal⁰ S⁰,¹ in
 /1          if⁰ S⁰ then emit⁰ O⁰ end ;⁰
             ⁰pause¹ ;¹
             emit¹ S¹
 /0       end signal⁰ ;⁰
          LCI2⁰
       end loop⁰
```

S^0 S^1

S^1

S^0

S^0

clef : $S^0 \neq S^1$

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène
4. Traiter la schizophrénie par duplication du corps
5. Plus efficace : réincarner la surface des boucles
6. Indexer les incarnations
7. Observer la guérison du pire des cas

```
output S1andS2, S1andnS2, nS1andS2, nS1andnS2 ;
loop
  trap T1 in
    signal S1 in
      pause ; emit S1 ; exit T1
    ||
      loop
        trap T2 in
          signal S2 in
            pause ; emit S2 ; exit T2
          ||
            loop
              if
                case (S1 and S2) do emit S1andS2
                case (S1 and not S2) do emit S1andnS2
                case (not S1 and S2) do emit nS1andS2
                case (not S1 and not) S2 do emit nS1andnS2
              end if ;
              pause
            end loop
          end signal
        end trap
      end loop
    end signal
  end trap
end loop
```

```
/0  output S1andS2, S1andnS2, nS1andS2, nS1andnS2 ;
    loop
      trap T1 in
/1      signal S1 in
          pause ; emit S1 ; exit T1
        ||
          loop
            trap T2 in
/2            signal S2 in
                pause ; emit S2 ; exit T2
              ||
                loop
                  if
                    case (S1 and S2) do emit S1andS2
                    case (S1 and not S2) do emit S1andnS2
                    case (not S1 and S2) do emit nS1andS2
                    case (not S1 and not) S2 do emit nS1andnS2
                  end if ;
                  pause
            end loop
/1        end signal
        end trap
      end loop
/0  end signal
    end trap
  end loop
```

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop
      trap T1^0 in
/1      signal S1^0,1 in
          pause ; emit S1^0,1 ; exit T1^0
        ||
          loop
            trap T2^0,1 in
/2            signal S2^0,1,2 in
                pause ; emit S2^0,1,2 ; exit T2^0,1
              ||
                loop
                  if
                    case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                    case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                    case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                    case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                  end if ;
                  pause
                end loop
/1            end signal
            end trap
          end loop
/0      end signal
      end trap
    end loop
```

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop
      trap T1^0 in
/1      signal S1^{0,1} in
          pause ; emit S1^{0,1} ; exit T1^0
        ||
          loop
            trap T2^{0,1} in
/2            signal S2^{0,1,2} in
                pause ; emit S2^{0,1,2} ; exit T2^{0,1}
              ||
                loop
                  if
                    case (S1^{0,1} and S2^{0,1,2}) do emit S1andS2^0
                    case (S1^{0,1} and not S2^{0,1,2}) do emit S1andnS2^0
                    case (not S1^{0,1} and S2^{0,1,2}) do emit nS1andS2^0
                    case (not S1^{0,1} and not S2^{0,1,2}) do emit nS1andnS2^0
                  end if ;
                  pause
                end loop
/1            end signal
            end trap
          end loop
/0      end signal
      end trap
    end loop
```

Matsya=0

Kurma=1

Varâha=2

```
/0   output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
     loop^0
       trap^0 T1^0 in
/1  ⟶  signal^0 S1^0,1 in
    ⟶    ^0pause ; emit S1^0,1 ; exit T1^0
    ⟶  ||^0
    ⟶    loop^0
    ⟶      trap^0 T2^0,1 in
/2  ⟶        signal^0 S2^0,1,2 in
    ⟶          ^0pause ; emit S2^0,1,2 ; exit T2^0,1
    ⟶        ||^0
    ⟶          loop^0
    ⟶            if^0
                    case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                    case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                    case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                    case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                  end if ;
                  pause
                end loop
/1            end signal
            end trap
          end loop
/0      end signal
      end trap
    end loop
```

instant 1a
index 0/0,0/1 et 0/2
quid du if ?

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop^0
      trap^0 T1^0 in
/1      signal^0 S1^0,1 in
          ^0pause ; emit S1^0,1 ; exit T1^0
        ||^0
          loop^0
            trap^0 T2^0,1 in
/2            signal^0 S2^0,1,2 in
                ^0pause ; emit S2^0,1,2 ; exit T2^0,1
              ||^0
                loop^0
                  if^0
                    case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                    case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                    case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                    case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                  end if^0 ;^0
                  ^0pause
                end loop
/1            end signal
            end trap
          end loop
/0      end signal
      end trap
    end loop
```

instant 1b
$S1^0$, $S2^0$ absents
$\Longrightarrow$ $nS1andnS2^0$
fin de l'instant

```
/0   output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
     loop^0
       trap^0 T1^0 in
/1       signal^0 S1^0,1 in
           ^0pause^1 ; emit S1^0,1 ; exit T1^0
         ||^0,1
           loop^0
             trap^0 T2^0,1 in
/2             signal^0 S2^0,1,2 in
                 ^0pause^2 ; emit S2^0,1,2 ; exit T2^0,1
               ||^0
                 loop^0
                   if^0
                     case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                     case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                     case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                     case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                   end if^0 ;^0
                   ^0pause^2
                 end loop
/1             end signal
             end trap
           end loop
/0       end signal
       end trap
     end loop
```

instant 2a
reprise des pause
index 1/1 = prof (0/1)
et 2/2 = prof (0/2)

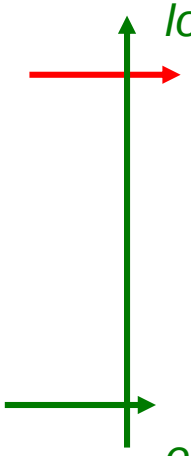
```
/0   output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
     loop^0
       trap^0 T1^0 in
/1       signal^0 S1^{0,1} in
           ^0pause^1 ; emit^1 S1^{0,1} ; exit T1^0
         ||^{0,1}
           loop^0
             trap^0 T2^{0,1} in
/2             signal^0 S2^{0,1,2} in
                 ^0pause^2 ; emit S2^{0,1,2} ; exit T2^{0,1}
               ||^0
                 loop^0
                   if^0
                     case (S1^{0,1} and S2^{0,1,2}) do emit S1andS2^0
                     case (S1^{0,1} and not S2^{0,1,2}) do emit S1andnS2^0
                     case (not S1^{0,1} and S2^{0,1,2}) do emit nS1andS2^0
                     case (not S1^{0,1} and not S2^{0,1,2}) do emit nS1andnS2^0
                   end if^0 ;^0
                   ^0pause^2
                 end loop
/1             end signal
             end trap
           end loop
/0       end signal
       end trap
     end loop
```

instant 2b,
branche 1 index 1/1
émission de $S1^1$

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop^0
      trap^0 T1^0 in
/1      signal^0 S1^{0,1} in
          ^0pause^1 ; emit^1 S1^{0,1} ; exit T1^0
        ||^{0,1}
          loop^0
            trap^0 T2^{0,1} in
/2            signal^0 S2^{0,1,2} in
                ^0pause^2 ; emit^2 S2^{0,1,2} ; exit T2^{0,1}
              ||^0
                loop^0
                  if^0
                    case (S1^{0,1} and S2^{0,1,2}) do emit S1andS2^0
                    case (S1^{0,1} and not S2^{0,1,2}) do emit S1andnS2^0
                    case (not S1^{0,1} and S2^{0,1,2}) do emit nS1andS2^0
                    case (not S1^{0,1} and not S2^{0,1,2}) do emit nS1andnS2^0
                  end if^0 ;^0
                  ^0pause^2
                end loop
/1            end signal
            end trap
          end loop
/0      end signal
      end trap
    end loop
```

instant 2c,
branche 2 index 2/2
émission de $S2^2$

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop^0
      trap^0 T1^0 in
/1      signal^0 S1^0,1 in
          ^0pause^1 ; emit^1 S1^0,1 ; exit T1^0
        ||^0,1
          loop^0
            trap^0 T2^0,1 in
/2            signal^0 S2^0,1,2 in
                ^0pause^2 ; emit^2 S2^0,1,2 ; exit T2^0,1
              ||^0,2
                loop^0,2
                  if^0,2
                    case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                    case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                    case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                    case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                  end if^0 ;^0
                  ^0pause^2
                end loop^2
/1            end signal
            end trap
          end loop
/0      end signal
      end trap
    end loop
```



instant 2d,
branche 3 index 2/2
rebouclage
quid du if ?

```
/0  output S1andS2⁰, S1andnS2⁰, nS1andS2⁰, nS1andnS2⁰ ;
    loop⁰
      trap⁰ T1⁰ in
/1      signal⁰ S1⁰,¹ in
          ⁰pause¹ ; emit¹ S1⁰,¹ ; exit T1⁰
        ||⁰,¹
          loop⁰
            trap⁰ T2⁰,¹ in
/2            signal⁰ S2⁰,¹,² in
                ⁰pause² ; emit² S2⁰,¹,² ; exit T2⁰,¹
              ||⁰,²
                loop⁰,²
                  if⁰,²
                    case (S1⁰,¹ and S2⁰,¹,²) do emit S1andS2⁰
                    case (S1⁰,¹ and not S2⁰,¹,²) do emit S1andnS2⁰
                    case (not S1⁰,¹ and S2⁰,¹,²) do emit nS1andS2⁰
                    case (not S1⁰,¹ and not S2⁰,¹,²) do emit nS1andnS2⁰
                  end if⁰,² ;⁰,²
                  ⁰,²pause²
                end loop²
/1            end signal
            end trap
          end loop
/0      end signal
      end trap
    end loop
```

instant 2e,
if en 2/2
$S1^1$ et $S2^2$ présents
$\Rightarrow$ $S1andS2^0$
et pause

```
/0   output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
     loop^0
       trap^0 T1^0 in
/1       signal^0 S1^0,1 in
           ^0pause^1 ; emit^1 S1^0,1 ; exit T1^0
         ||^0,1
           loop^0
             trap^0 T2^0,1 in
/2             signal^0 S2^0,1,2 in
                 ^0pause^2 ; emit^2 S2^0,1,2 ; exit^2 T2^0,1
               ||^0,2
                 loop^0,2
                   if^0,2
                     case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                     case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                     case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                     case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                   end if^0,2 ;^0,2
                   ^0,2pause^2
                 end loop^2
/1             end signal
             end trap^1
           end loop
/0       end signal
       end trap
     end loop
```

instant 2f,
sortie de T2[1] (pop)
mort de son corps

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop^0
      trap^0 T1^0 in
/1      signal^0 S1^{0,1} in
          ^0pause^1 ; emit^1 S1^{0,1} ; exit T1^0
        ||^{0,1}
          loop^{0,1}
            trap^{0,1} T2^{0,1} in
/2            signal^{0,1} S2^{0,1,2} in
                ^{0,1}pause^2 ; emit^2 S2^{0,1,2} ; exit^1 T2^{0,1}
              ||^{0,1,2}
                loop^{0,1,2}
                  if^{0,1,2}
                    case (S1^{0,1} and S2^{0,1,2}) do emit S1andS2^0
                    case (S1^{0,1} and not S2^{0,1,2}) do emit S1andnS2^0
                    case (not S1^{0,1} and S2^{0,1,2}) do emit nS1andS2^0
                    case (not S1^{0,1} and not S2^{0,1,2}) do emit nS1andnS2^0
                  end if^{0,2} ;^{0,2}
                  ^{0,1}pause^2
                end loop
/1            end signal
            end trap^1
          end loop^1
/0      end signal
      end trap
    end loop
```

instant 2g, rebouclage comme instant 1b mais avec index 1

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop^0
      trap^0 T1^0 in
/1      signal^0 S1^{0,1} in
          ^0pause^1 ; emit^1 S1^{0,1} ; exit T1^0
        ||^{0,1}
          loop^{0,1}
            trap^{0,1} T2^{0,1} in
/2            signal^{0,1} S2^{0,1,2} in
                ^{0,1}pause^2 ; emit^2 S2^{0,1,2} ; exit^1 T2^{0,1}
              ||^{0,1,2}
                loop^{0,1,2}
                  if^{0,1,2}
                    case (S1^{0,1} and S2^{0,1,2}) do emit S1andS2^0
                    case (S1^{0,1} and not S2^{0,1,2}) do emit S1andnS2^0
                    case (not S1^{0,1} and S2^{0,1,2}) do emit nS1andS2^0
                    case (not S1^{0,1} and not S2^{0,1,2}) do emit nS1andnS2^0
                  end if^{0,2} ;^{0,1,2}
                  ^{0,1,2}pause^2
                end loop^{0,1,2}
/1            end signal
            end trap^1
          end loop^1
/0      end signal
      end trap
    end loop
```

instant 2h,
$S1^1$ présent
$S2^1$ absent
$\Longrightarrow$ $S1andnS2^0$

```
/0   output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
     loop^0
       trap^0 T1^0 in
/1       signal^0 S1^0,1 in
           0pause^1 ; emit^1 S1^0,1 ; exit^1 T1^0
         ||^0,1
           loop^0,1
             trap^0,1 T2^0,1 in
/2             signal^0,1 S2^0,1,2 in
                 0,1pause^2 ; emit^2 S2^0,1,2 ; exit^1 T2^0,1
               ||^0,1,2
                 loop^0,1,2
                   if^0,1,2
                     case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                     case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                     case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                     case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                   end if^0,2 ;^0,1,2
                   0,1,2pause^2
                 end loop^0,1,2
/1             end signal
             end trap^1
           end loop^1
/0       end signal
       end trap^0
     end loop
```

instant 2i,
sortie de $T1^0$ (pop (1/1))
mort de son corps
sortie de signal S1 et S2

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop^0
      trap^0 T1^0 in
/1      signal^0 S1^0,1 in
          0pause^1 ; emit^1 S1^0,1 ; exit^1 T1^0
        ||^0,1
          loop^0,1
            trap^0,1 T2^0,1 in
/2            signal^0,1 S2^0,1,2 in
                0,1pause^2 ; emit^2 S2^0,1,2 ; exit^1 T2^0,1
              ||^0,1,2
                loop^0,1,2
                  if^0,1,2
                    case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                    case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                    case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                    case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                  end if^0,2 ;^0,1,2
                  0,1,2pause^2
                end loop^0,1,2
/1            end signal
            end trap^1
          end loop^1
/0      end signal
      end trap^0
    end loop^0
```

instant 2j,
rebouclage global,
idem premier instant
$S1^0$ et $S2^0$ absents
$\Longrightarrow$ $nS1andnS2^0$

```
/0  output S1andS2^0, S1andnS2^0, nS1andS2^0, nS1andnS2^0 ;
    loop^0
      trap^0 T1^0 in
/1      signal^0 S1^0,1 in
          ^0pause^1 ; emit^1 S1^0,1 ; exit^1 T1^0
        ||^0,1
          loop^0,1
            trap^0,1 T2^0,1 in
/2            signal^0,1 S2^0,1,2 in
                ^0,1pause^2 ; emit^2 S2^0,1,2 ; exit^1 T2^0,1
              ||^0,1,2
                loop^0,1,2
                  if^0,1,2
                    case (S1^0,1 and S2^0,1,2) do emit S1andS2^0
                    case (S1^0,1 and not S2^0,1,2) do emit S1andnS2^0
                    case (not S1^0,1 and S2^0,1,2) do emit nS1andS2^0
                    case (not S1^0,1 and not S2^0,1,2) do emit nS1andnS2^0
                  end if^0,2 ;^0,1,2
                   ^0,1,2pause^2
                end loop^0,1,2
/1            end signal
            end trap^1
          end loop^1
/0      end signal
      end trap^0
    end loop^0
```

instant 2j,
rebouclage global,
idem premier instant
$S1^0$ et $S2^0$ absents
$\Longrightarrow$ $nS1andnS2^0$

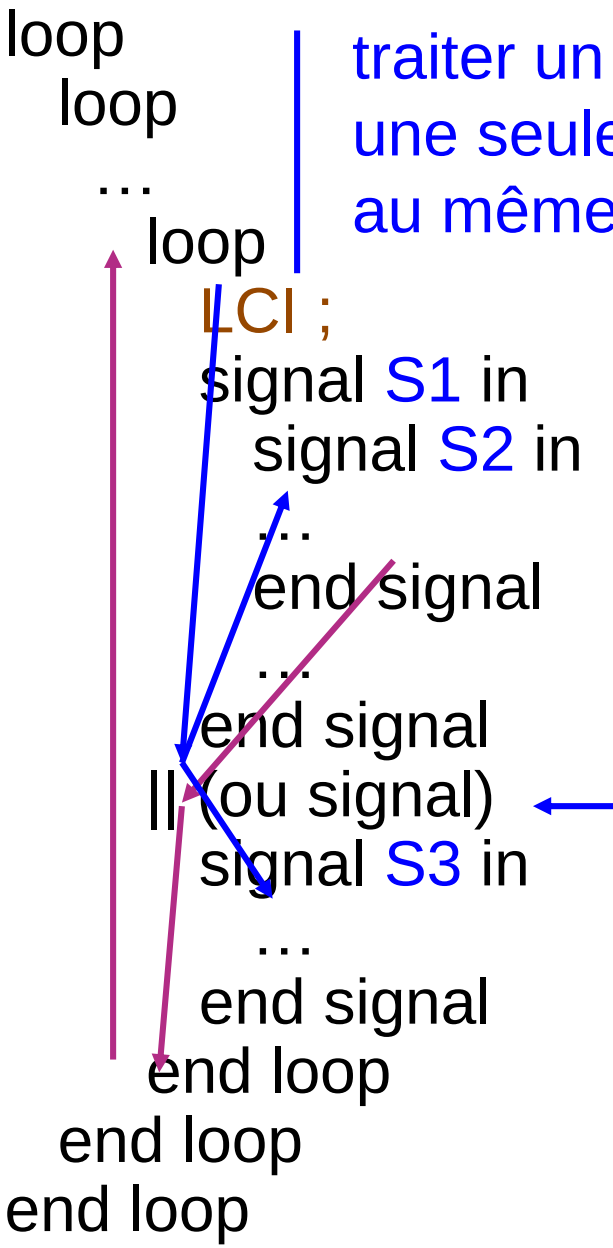
Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène
4. Traiter la schizophrénie par duplication du corps
5. Plus efficace : réincarner la surface des boucles
6. Indexer les incarnations
7. Observer la guérison du pire des cas
8. Optimisations
9. Tardieu fait il aussi bien en plus algébrique ?

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène
4. Traiter la schizophrénie par duplication du corps
5. Plus efficace : réincarner la surface des boucles
6. Indexer les incarnations
7. Observer la guérison du pire des cas
8. Optimisations

Eviter les duplications inutiles



traiter un paquet de boucles comme
une seule boucle (deux ne peuvent pas boucler
au même instant, sinon boucle instantanée)

Schizophénie si et seulement
si il existe un chemin direct
de la sortie du || (ou signal)
supérieur vers sa propre entrée

Eviter les duplications inutiles

Résultat : la duplication est quasi-linéraire en pratique
(indispensable pour les vrais programmes)

L'optimisation algébrique de Tardieu

$$
\begin{aligned}
dup4(K, \texttt{nothing}) &\stackrel{def}{=} \texttt{nothing} \\
dup4(K, \mathit{label}\texttt{:pause}) &\stackrel{def}{=} \mathit{label}\texttt{:pause} \\
dup4(K, \texttt{exit } T) &\stackrel{def}{=} \texttt{exit } T \\
dup4(K, \texttt{emit } S) &\stackrel{def}{=} \texttt{emit } S \\
dup4(K, \texttt{present ... end}) &\stackrel{def}{=} \texttt{present } S \texttt{ then } dup4(K, p) \texttt{ else } dup4(K, q) \texttt{ end} \\
dup4(K, \texttt{loop } p \texttt{ end}) &\stackrel{def}{=} \texttt{loop } dup4(K \cup \{0\}, p) \texttt{ end} \\
dup4(K, \texttt{trap } T \texttt{ in } p \texttt{ end}) &\stackrel{def}{=} \texttt{trap } T \texttt{ in } dup4(\{k \in \mathbb{N}, \downarrow k \in K\}, p) \texttt{ end} \\
dup4(K, p; q) &\stackrel{def}{=} \begin{bmatrix} dup4((\text{if } K \cap \Gamma_q = \emptyset \text{ then } K \backslash \{0\} \text{ else } K \cup \{0\}), p); \\ dup4((\text{if } 0 \in \Gamma_p \text{ then } K \text{ else } \emptyset), q) \end{bmatrix} \\
dup4(K, \texttt{signal } S \texttt{ in } p \texttt{ end}) &\stackrel{def}{=} \begin{bmatrix} \text{if} & K \cap \Omega_p = \emptyset \\ \text{then} & \texttt{signal } S \texttt{ in } dup4(\emptyset, p) \texttt{ end} \\ \text{else} & \begin{bmatrix} \texttt{signal } S \texttt{ in } surf(p) \texttt{ end}; \\ skip(\texttt{signal } S \texttt{ in } dup4(\emptyset, p) \texttt{ end}) \end{bmatrix} \end{bmatrix} \\
dup4(K, p \parallel q) &\stackrel{def}{=} \begin{bmatrix} \text{if} & K \cap \Omega_{p \parallel q} = \emptyset \\ \text{then} & dup4(\emptyset, p) \parallel dup4(\emptyset, q) \\ \text{else} & \{surf(p) \parallel surf(q)\}; skip(\{dup4(\emptyset, p) \parallel dup4(\emptyset, q)\}) \end{bmatrix}
\end{aligned}
$$

Conjecture

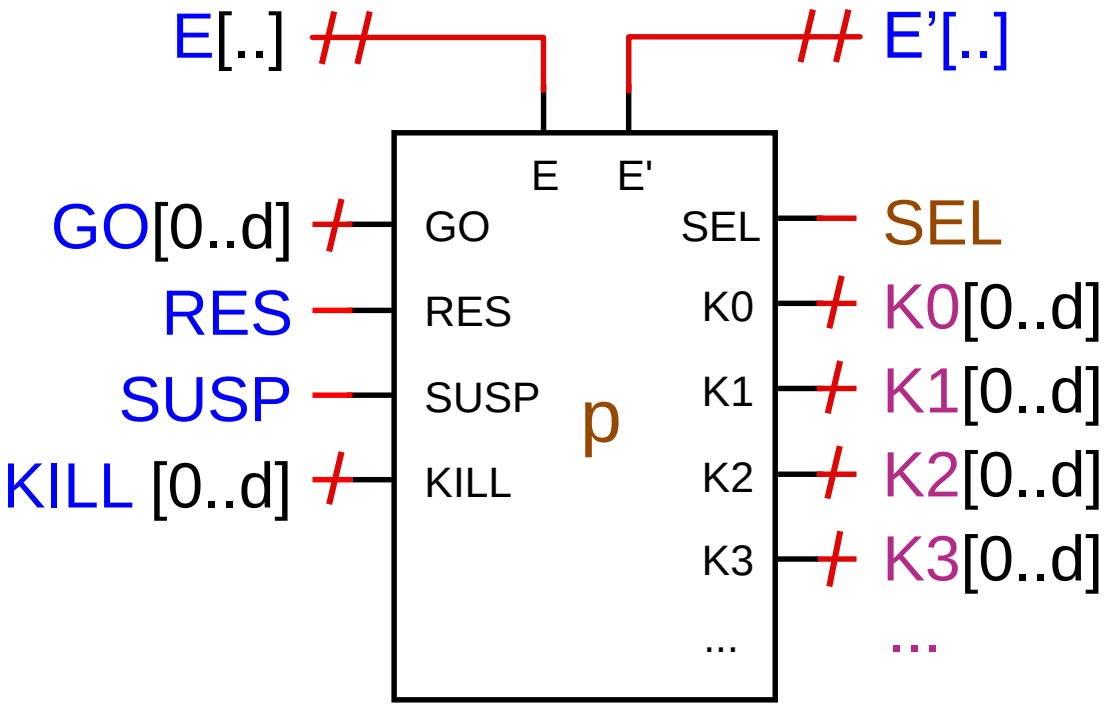
Les traitements de la schizophrénie par
GB et Tardieu donnent les mêmes duplications

=> introduire les index d'incarnation dans dup4 !

Agenda

1. Pourquoi une étude spéciale des boucles
2. Un programme schizophrène
3. Un programme doublement schizophrène
4. Traiter la schizophrénie par duplication du corps
5. Plus efficace : réincarner la surface des boucles
6. Indexer les incarnations
7. Observer la guérison du pire des cas
8. Optimisations
9. La réincarnation dans les circuits

Travailler avec des faisceaux de fils !



dans p | non schizophrène : [0..d]
schizophrène : [0..d+1]

Transformer les portes en vecteurs de portes

Cablâge de faisceaux dans le cas p schizo

push

Pour X dans GO, KILL, S

X[d..d] → Xp[d..d+1] : X[i] → Xp[i] pour tout i ≤ d
Xp[d+1] non câblé

Pour Y dans les Ki

pop

Y[0..d+1] → Yp[0..d] : Y[i] → Yp[i] pour tout i ≤ d
Y[d+1] → Yp[d]

Câblage final de pause

Pour X dans GO, KILL, S

push

X[d..d] → Xp[d..d+1] : X[i] → Xp[i] pour tout i ≤ d
Xp[d+1] non câblé

Pour Y dans les Ki

pop

Y[0..d+1] → Yp[0..d] : Y[i] → Yp[i] pour tout i ≤ d
Y[d+1] → Yp[d]

Conclusion

- Traiter la schizophrénie est indispensable pour compiler des programmes de grande taille
- Les méthodes GB et Tardieu conduisent à des expansions sublinéaires, et sont peut-être identiques
- Et les optimisations BDD améliorent beaucoup les circuits, voir cours 6