

Modellierung nebenläufiger Systeme

Systeme, in welchen Komponenten parallel arbeiten:

1. Technische Systeme
2. Softwaresysteme
3. ...

Komponenten arbeiten teilweise

- ▶ unabhängig
- ▶ abhängig

voneinander.

Beispiel

Einfaches Programm:

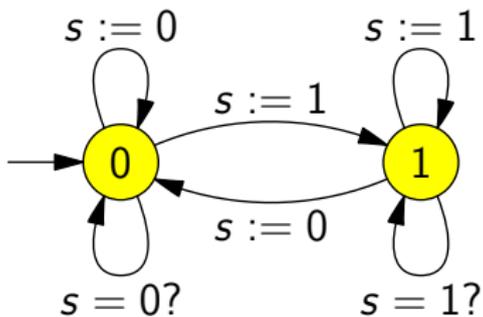
```
s := 0;  
s := 1;  
if (s=0) print ;  
else exit ;
```

Zerlegen in Komponenten:

- ▶ Kontrollstruktur (welche Anweisung wird ausgeführt?)
- ▶ Inhalt der Variablen

Interaktion, falls Folgeanweisung vom Variableninhalt abhängt.

Modellierung einer booleschen Variable:

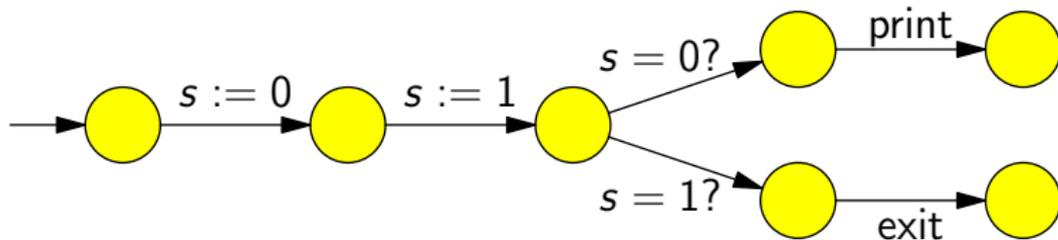


Mögliche Aktionen:

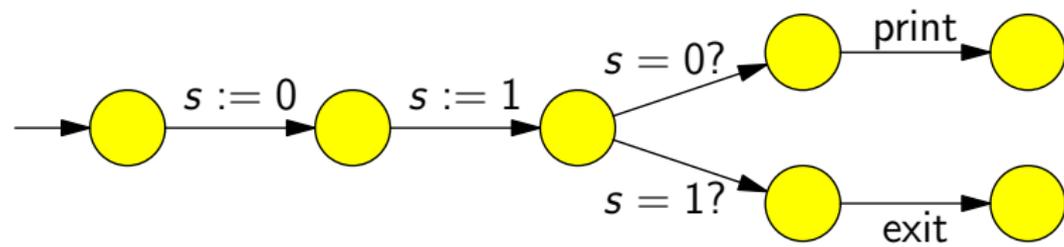
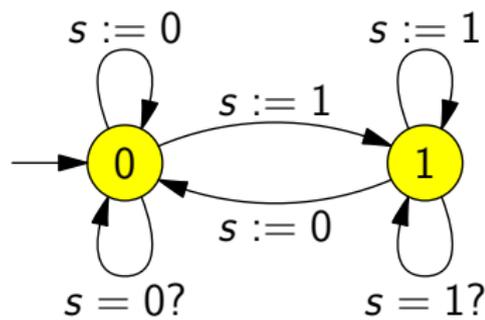
- $s := 0$ Variable auf 0 setzen
- $s := 1$ Variable auf 1 setzen
- $s = 0?$ Variable auf 0 testen
- $s = 1?$ Variable auf 1 testen

```
s := 0;  
s := 1;  
if (s=0) print;  
else exit ;
```

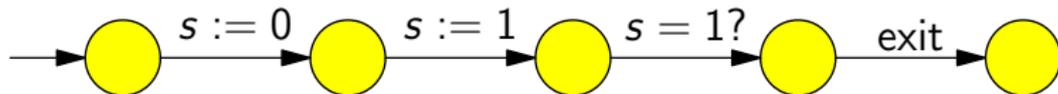
Modellierung der Kontrollstruktur:



Synchronisiertes Produkt



$B \circ K:$



Synchronisiertes Produkt

Definition

Es seien $M' = (\Sigma', Q', \delta', q'_0, F')$ und $M'' = (\Sigma'', Q'', \delta'', q''_0, F'')$ zwei NFA's.

Wir definieren das *synchronisierte Produkt* $M = M' \circ M''$:

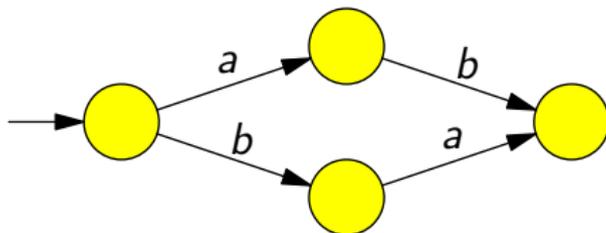
$M = (\Sigma, Q' \times Q'', \delta, (q'_0, q''_0), F' \times F'')$ mit $\Sigma = \Sigma' \cup \Sigma''$ und

- ▶ $(q', p') \in \delta((q, p), a)$ falls $a \in \Sigma' \cap \Sigma''$ und $q' \in \delta'(q, a)$, $p' \in \delta''(p, a)$.
- ▶ $(q', p) \in \delta((q, p), a)$ falls $a \in \Sigma' \setminus \Sigma''$ und $q' \in \delta'(q, a)$.
- ▶ $(q, p') \in \delta((q, p), a)$ falls $a \in \Sigma'' \setminus \Sigma'$ und $p' \in \delta''(p, a)$.

Beispiel



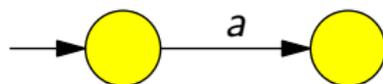
Was ist das synchronisierte Produkt?



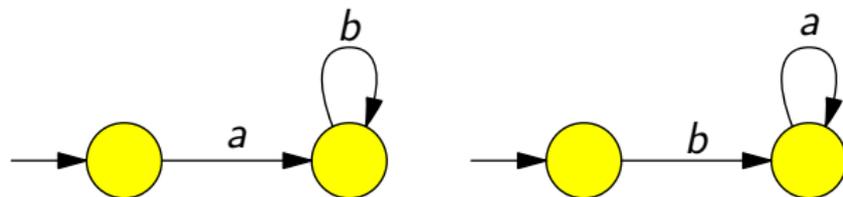
Beispiel



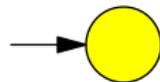
Was ist das synchronisierte Produkt?



Beispiel



Was ist das synchronisierte Produkt?



Unsynchronisiertes Produkt

Definition

Es seien $M' = (\Sigma', Q', \delta', q'_0, F')$ und $M'' = (\Sigma'', Q'', \delta'', q''_0, F'')$ zwei NFA's.

Wir definieren das *unsynchronisierte Produkt* $M = M' \sqcup M''$:

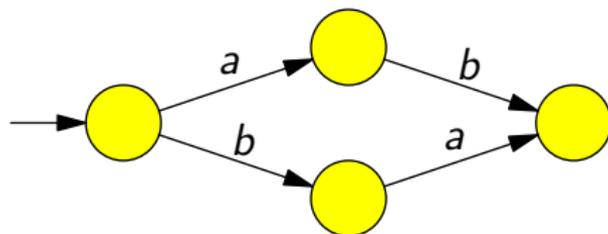
$M = (\Sigma, Q' \times Q'', \delta, (q'_0, q''_0), F' \times F'')$ mit $\Sigma = \Sigma' \cup \Sigma''$ und

- ▶ $(q', p) \in \delta((q, p), a)$ falls $a \in \Sigma'$ und $q' \in \delta'(q, a)$.
- ▶ $(q, p') \in \delta((q, p), a)$ falls $a \in \Sigma''$ und $p' \in \delta''(p, a)$.

Beispiel



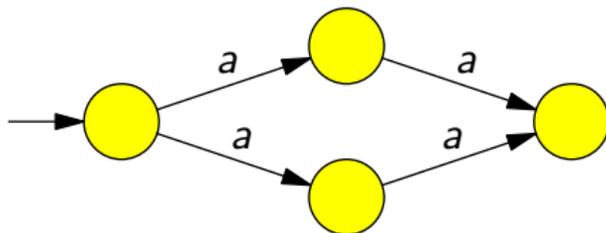
Was ist das unsynchronisierte Produkt?



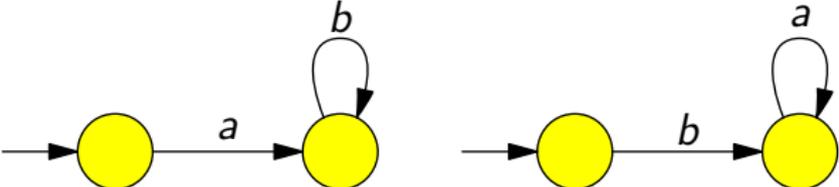
Beispiel



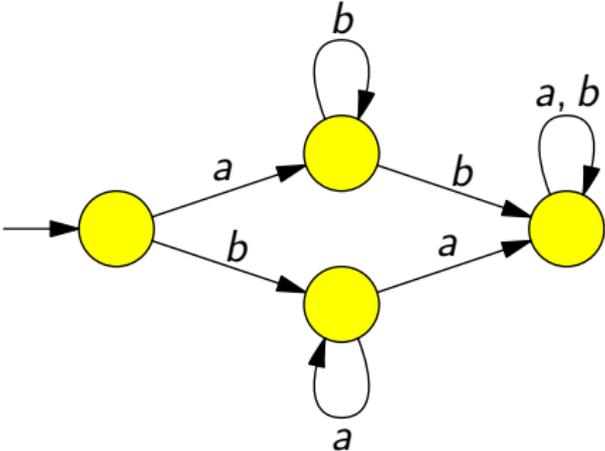
Was ist das unsynchronisierte Produkt?



Beispiel



Was ist das unsynchronisierte Produkt?



Das Mutual-Exclusion-Problem (Mutex)

```
while(true) {  
    /* non critical */  
    u:=1;  
    while(x=1) { }  
    /* critical section */  
    u:=0;  
}
```

```
while(true) {  
    /* non critical */  
    x:=1;  
    while(u=1) { }  
    /* critical section */  
    x:=0;  
}
```

Nur ein Programm darf sich in der *critical section* befinden.

Modellierung des Prozesses P_0 :

```
while(true) {  
  /* non critical */  
  u:=1;  
  while(x=1) { }  
  /* critical section */  
  u:=0;  
}
```

