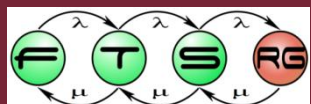


Adatfolyam hálók

Dr. Bartha Tamás, Dr. Pataricza András főlektori



Adatfolyam modellezés

Nem determinisztikus DFN formalizmus

- [Jonsson, Cannata]
- **Struktúra**
 - Adatfolyam gráf (DFG)
 - csomópontok
 - irányított élek (FIFO csatornák)
- **Viselkedés**
 - Tüzelési szabályok: $\langle 0; in=0; 1; out=2; \pi \rangle$
- **Adatok**
 - Tokenek

A módszer előnyei

Tulajdonság	Alkalmas
Grafikus, moduláris, kompakt, hierarchikus	Egyszerűen áttekinthető modell
Fekete és átlátszó doboz modell	Modellezés korai fázisban
Finomítási szabályok	Többszintű modellezés
Információáramlás direkt leírása	Hibaterjedés modellezése
Elosztott modell mind finom, mind durva pontossággal	Aszinkron, konkurens események
Adatvezérelt működés	Eseményvezérelt real-time rendszerek
Hívási átlátszóság, atomi tulajdonság, információrejtés	Hibatűrő alkalmazások
Matematikai formalizmus	Formális módszerek
Transzformáció: TTPN, PA	Validáció, időbeli analízis

Adatfolyam hálózat formális leírása

- **Adatfolyam hálózat:** egy hármás (N, C, S)

- N : csomópontok halmaza

- C : csatornák halmaza

- I: bemenő csatornák

- O: kimenő csatornák

- IN: belső (csomópontok közötti) csatornák

} kapcsolat a külvilággal

- S : állapotok halmaza

- **Adatfolyam csatorna:**

- végtelen kapacitású FIFO csatorna,

- egy bemeneti és egy kimeneti csomóponthoz kötve

- állapota: $S_c = \times^\infty M_c$ tokenszekvencia

Adatfolyam csomópont formális leírása

Adatfolyam csomópont: $n = (I_n, O_n, S_n, s_n^0, R_n, M_n)$, ahol

I_n – bemenő csatornák halmaza

O_n – kimenő csatornák halmaza

S_n – csomópont állapotok halmaza

s_n^0 – csomópont kezdőállapota, $s_n^0 \in S_n$

M_n – tokenek halmaza

R_n – tüzelések halmaza, $r_n \in R_n$ egy ötös $(s_n, X_{in}, s'_n, X_{out}, \pi)$

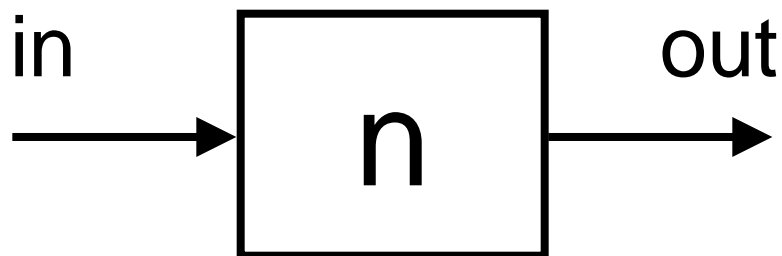
S_n – tüzelés előtti és utáni állapotok, $s'_n \in S$

X_{in} – bemenő leképezés, $X_{in} : I_n \rightarrow M_n$

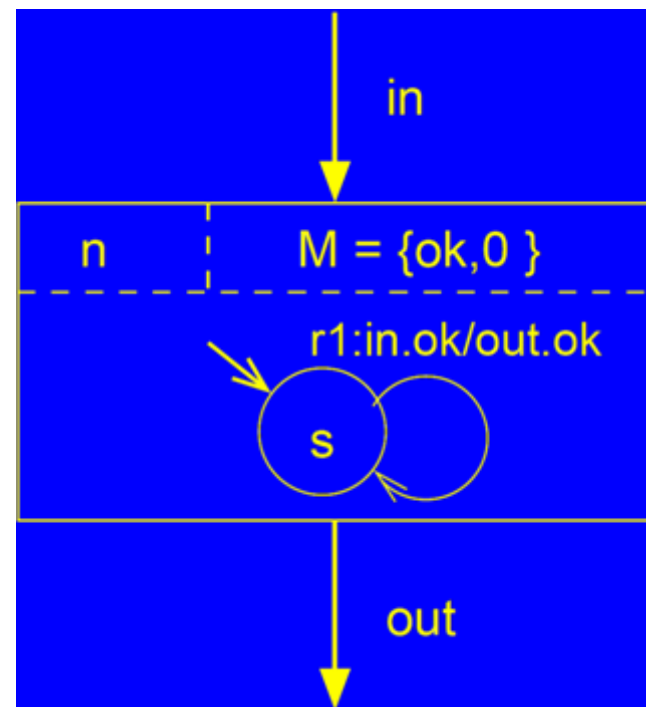
X_{out} – kimenő leképezés, $X_{out} : O_n \rightarrow M_n$

π – tüzelés prioritása, $\pi \in N$

Egy példa



- Egy token kapacitású csatornák
- Hálózat:
 - DFN = ($\{n\}$, $\{in, out\}$,
 - $\{(s,0,0), (s,ok,0), (s,0,ok), (s,ok,ok)\}$)
- Csomópontok:
 - $n = (\{in\}, \{out\}, \{s\}, s, \{ok,0\}, \{r1\})$
- Tüzelések:
 - $r1 = \langle s; in=ok; s; out=ok; 0 \rangle$



Adatfolyam modellek kiértékelése

- + Interaktív szimuláció
- Validáció, helyességbizonyítás (direkt/indirekt)
 - Dinamikus tulajdonságok: elérhetőség, holtpontmentesség
- + Időbeli analízis (indirekt)
 - Tüzelési szabályokban végrehajtási idő, mint valószínűségi változó
- + Hibaszimuláció (direkt, diszkrét esemény szimuláció)
 - Működési modell kiegészítése hibamoddellel, hibahatások elemzése
- + Teszttervezés (indirekt)
 - Tesztgenerálás, tesztelhetőségi analízis, tesztkészlet optimalizálás
- Hibahatás analízis (direkt)
 - FMEA: hibamód és hatás analízis, hibafa és eseményfa generálás
- (Megbízhatósági analízis) (indirekt)
 - Klasszikus mértékek: megbízhatóság, rendelkezésre állás, MTBF, ...

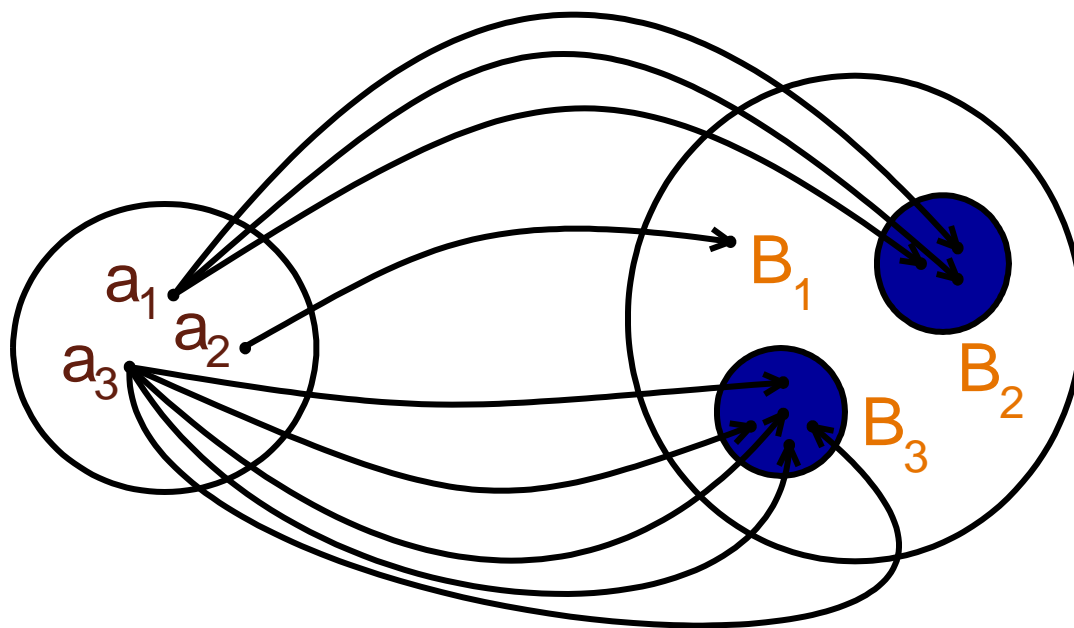
Modellfinomítás adatfolyam hálókra

Modellfinomítás:

- Többszintű modellezés
- Szintek közötti átjárás (formalizált szabályok)
- Állapot és viselkedési konzisztencia megőrzése

Halmazfinomítás

Diszjunkt részhalmazok hozzárendelése elemekhez



$\forall a_i, \in A, R(a_i) \subset B$ úgy, hogy $R(a_i) \cap R(a_j) = \emptyset \forall i, j$

Modellfinomítás adatfolyam hálókra

■ Fekete doboz nézet

- Csak a környezettel való kapcsolat jelenik meg
 - szintaktikus interfész: ki- és bemeneti csatornák, üzenettípusok
 - szemantikus interfész: ki- és bemenő üzenetek kapcsolata

■ Átlátszó doboz nézet

- Kommunikáció finomítás
 - komponens szintaktikus interfészének megváltoztatása
 - ki- és bemeneti csatornák és az üzenettípusok száma változik
- Állapottér finomítás
 - állapotátmeneti reláció elemeinek száma
- Eloszlás finomítás
 - dekompozíció, részkomponensekre bontás

Modellfinomítás adatfolyam hálókra

Fekete- és átlátszó dobozokra megfogalmazott elvek általánosítása adatfolyam hálókra:

- **Értelmezési tartomány finomítás**
 - Token halmaz finomítás
 - Állapothalmaz finomítás
- **Struktúra finomítás**
 - Csomópont helyettesítése adatfolyam alhálóval

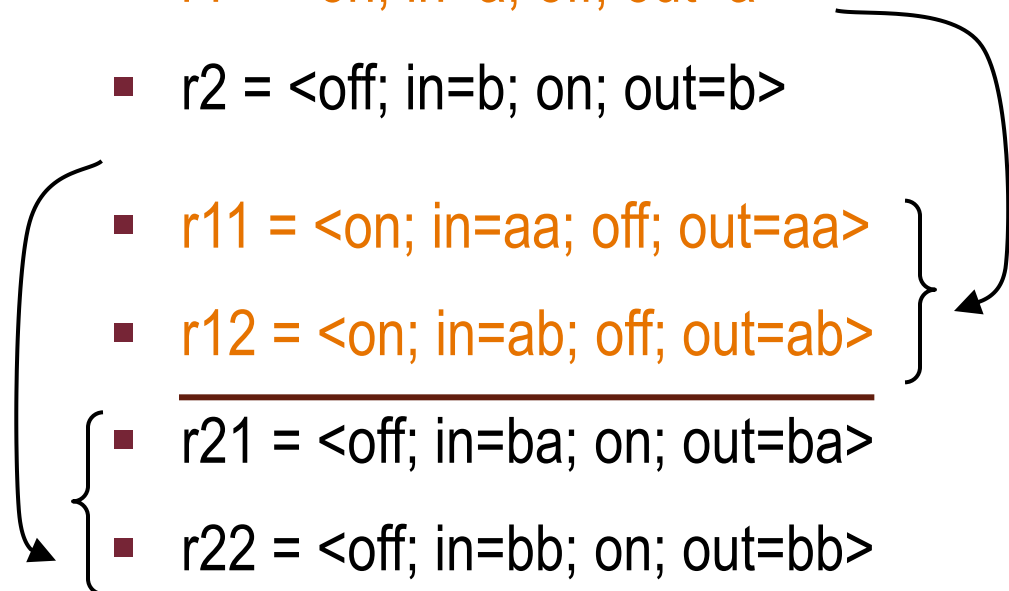
Értelmezési tartomány finomítás

- **Értelmezési tartomány finomítás**
 - Tokenek halmazának finomítása: M'_n finomítottja M_n -nek
 - Állapotok halmazának finomítása: S'_n finomítottja S_n -nek
- Ki- és bemeneti csatornák változatlanok
- Tüzelési szabályok megfelelő változtatása

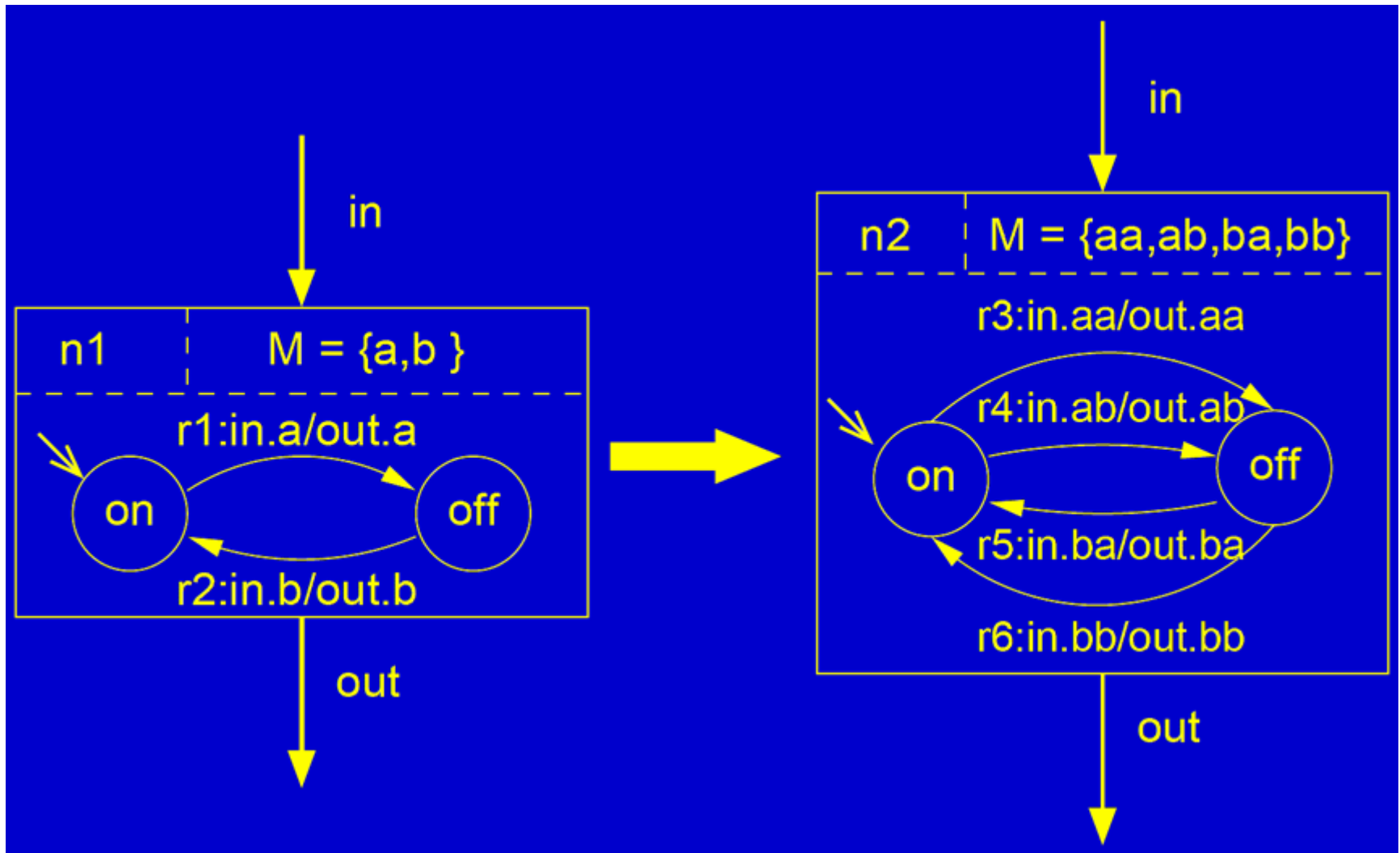
Tokenek halmazának finomítása: példa

	n_1	$n_2 = \mathfrak{R}(n_1)$
Állapotok	on off	{on} {off}
Tokenek	a b	{aa, ab} {ba, bb}
Tüzelési szabályok	r1 r2	{r11, r12} {r21, r22}

- $r1 = \langle \text{on}; \text{in}=\text{a}; \text{off}; \text{out}=\text{a} \rangle$
- $r2 = \langle \text{off}; \text{in}=\text{b}; \text{on}; \text{out}=\text{b} \rangle$
- $r11 = \langle \text{on}; \text{in}=\text{aa}; \text{off}; \text{out}=\text{aa} \rangle$
- $r12 = \langle \text{on}; \text{in}=\text{ab}; \text{off}; \text{out}=\text{ab} \rangle$
- $r21 = \langle \text{off}; \text{in}=\text{ba}; \text{on}; \text{out}=\text{ba} \rangle$
- $r22 = \langle \text{off}; \text{in}=\text{bb}; \text{on}; \text{out}=\text{bb} \rangle$



Értelmezési tartomány finomítás: tokenek

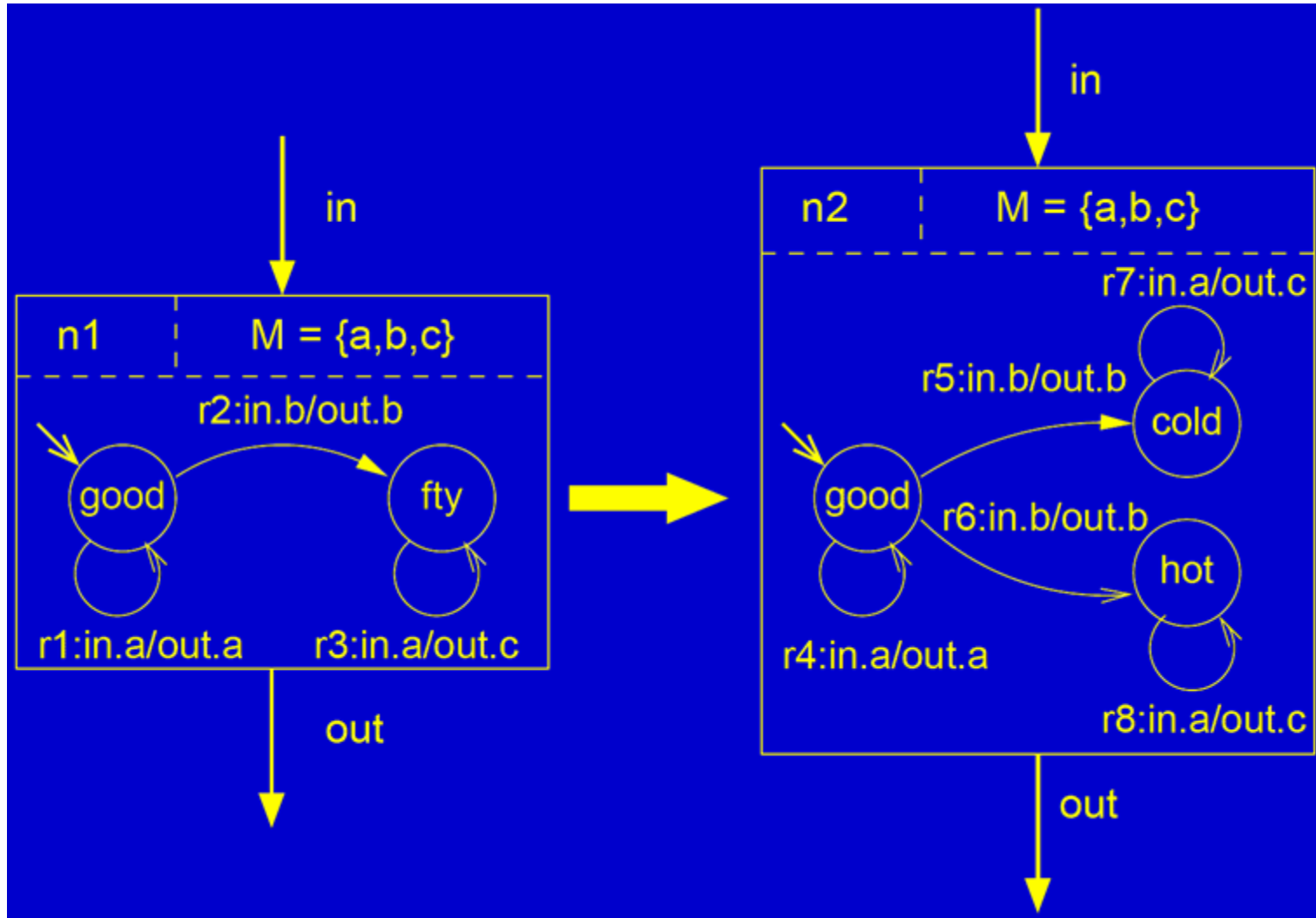


Állapotok halmazának finomítása: példa

	n_1	$n_2 = \mathcal{R}(n_1)$
Állapotok	good	{good}
	fty	{hot, cold}
Tokenek	a	{a}
	b	{b}
	c	{c}
Tüzelési szabályok	r1	{r11}
	r2	{r21, r22}
	r3	{r31, r32}

- $r1 = \langle \text{good}; \text{in}=a; \text{good}; \text{out}=a \rangle$
- $r2 = \langle \text{good}; \text{in}=b; \text{fty}; \text{out}=b \rangle$
- $r3 = \langle \text{fty}; \text{in}=a; \text{fty}; \text{out}=c \rangle$
- $r11 = \langle \text{good}; \text{in}=a; \text{good}; \text{out}=a \rangle$
- $r21 = \langle \text{good}; \text{in}=b; \text{cold}; \text{out}=b \rangle$
- $r22 = \langle \text{good}; \text{in}=b; \text{hot}; \text{out}=b \rangle$
- $r31 = \langle \text{cold}; \text{in}=a; \text{cold}; \text{out}=c \rangle$
- $r32 = \langle \text{hot}; \text{in}=a; \text{hot}; \text{out}=c \rangle$

Értelmezési tartomány finomítás: állapotok



Példa: referenciajel-generátor



- Hibamodell:

OK – névleges feszültség

FTY – névlegestől eltérő feszültség

- Működés:

r0 = <s0; power_in=OK; s0; ref_out=OK>

r1 = <s0; power_in=FTY; s0; ref_out=OK>

r2 = <s0; power_in=FTY; s1; ref_out=FTY>

r3 = <s1; power_in=OK; s1; ref_out=FTY>

r4 = <s1; power_in=FTY; s1; ref_out=FTY>

Példa: referenciajel-generátor (finomított működés)

1. Állapothalmaz finomítás: $s1 \rightarrow s1a, s1b$

$r0 = \langle s0; \text{power_in} = \text{OK}; s0; \text{ref_out} = \text{OK} \rangle$

$r1 = \langle s0; \text{power_in} = \text{FTY}; s0; \text{ref_out} = \text{OK} \rangle$

$r2 = \langle s0; \text{power_in} = \text{FTY}; s1a; \text{ref_out} = \text{FTY} \rangle$

$r31 = \langle s1a; \text{power_in} = \text{OK}; s1a; \text{ref_out} = \text{FTY} \rangle$

$r32 = \langle s1b; \text{power_in} = \text{OK}; s1b; \text{ref_out} = \text{FTY} \rangle$

$r41 = \langle s1a; \text{power_in} = \text{FTY}; s1b; \text{ref_out} = \text{FTY} \rangle$

$r42 = \langle s1b; \text{power_in} = \text{FTY}; s1b; \text{ref_out} = \text{FTY} \rangle$

2. Tokenfinomítás: $\text{FTY} \rightarrow \text{LOW}, \text{HIGH}$ (s0 állapot)

Tokenfinomítás: $\text{FTY} \rightarrow \text{LOW}, \text{HIGH}$ (s1 állapot)

Példa: referenciajel-generátor (finomított működés)

1. **Állapothalmaz finomítás:** $s1 \rightarrow s1a, s1b$
2. **Tokenfinomítás:** $FTY \rightarrow \text{LOW}, \text{HIGH}$ (s0 állapot)
r0=<s0; power_in=OK; s0; ref_out=OK>
r11=<s0; power_in=LOW; s0; ref_out=OK>
r21=<s0; power_in=HIGH; s1a; ref_out=HIGH>
r31=<s1a; power_in=OK; s1a; ref_out=FTY>
r32=<s1b; power_in=OK; s1b; ref_out=FTY>
r41=<s1a; power_in=FTY; s1b; ref_out=FTY>
r42=<s1b; power_in=FTY; s1b; ref_out=FTY>
3. **Tokenfinomítás:** $FTY \rightarrow \text{LOW}, \text{HIGH}$ (s1 állapot)

Példa: referenciajel-generátor (finomított működés)

1. **Állapothalmaz finomítás:** $s1 \rightarrow s1a, s1b$
2. **Tokenfinomítás:** FTY \rightarrow LOW, HIGH (s0 állapot)
3. **Tokenfinomítás:** FTY \rightarrow **LOW**, **HIGH** (s1 állapot)

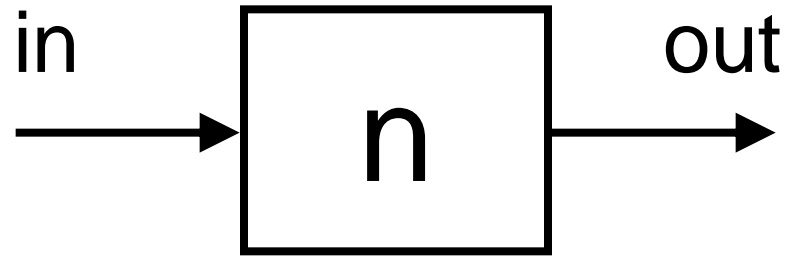
r0=<s0; power_in=OK; s0; ref_out=OK>
r11=<s0; power_in=LOW; s0; ref_out=LOW>
r21=<s0; power_in=HIGH; s1a; ref_out=HIGH>
r311=<s1a; power_in=OK; s1a; ref_out=LOW>
r321=<s1b; power_in=OK; s1b; ref_out=HIGH>
r411=<s1a; power_in=LOW; s1b; ref_out=LOW>
r412=<s1a; power_in=HIGH; s1b; ref_out=LOW>
r421=<s1b; power_in=LOW; s1b; ref_out=LOW>
r422=<s1b; power_in=HIGH; s1b; ref_out=HIGH>

Bizonytalanság megszűnt

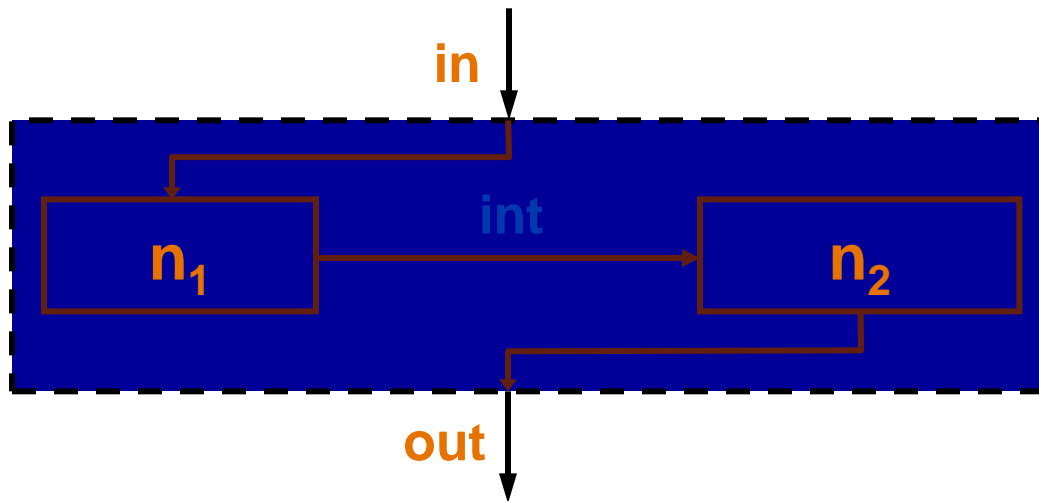
Struktúra finomítás

- **Struktúra finomítás:**
 - Struktúra módosítás
 - környezethez kapcsolódó csatornák változatlanok
 - belső csomópontok és csatornák keletkeznek
 - Állapotmegfeleltetés: csomópont \leftrightarrow részháló
 - Tokenek halmaza változatlan
 - Tüzelésekből *tüzelési szekvenciák* megfelelő kialakítása

Struktúra finomítás: példa



↓ DFN = $\mathfrak{R}(n)$



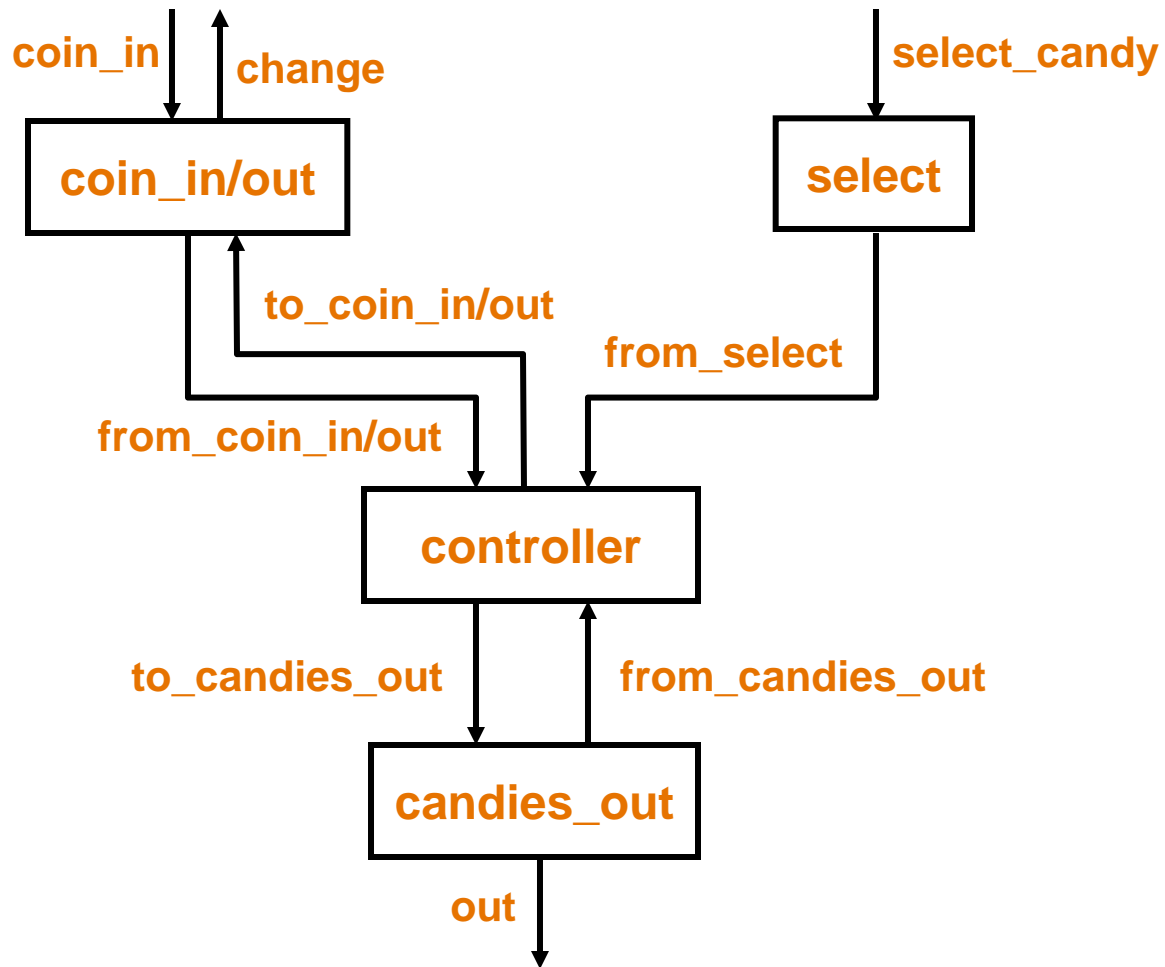
Struktúra finomítás: példa

	n_1	$n_2 = \mathfrak{R}(n_1)$
Állapotok	good fty	$\{\{\text{good, good, X}\},$ $\{\text{good, fty, X}\}\}$ $\{\{\text{fty, good, X}\},$ $\{\text{fty, fty, X}\}\}$
Tokenek	a b	$\{\text{a}\}$ $\{\text{b}\}$
Tüzelési szabályok	r1 r2	$\{\langle r_{n1}1; r_{n2}1 \rangle;$ $\langle r_{n1}1; r_{n2}3 \rangle\}$ $\{\langle r_{n1}2; r_{n2}2 \rangle;$ $\langle r_{n1}2; r_{n2}4 \rangle\}$

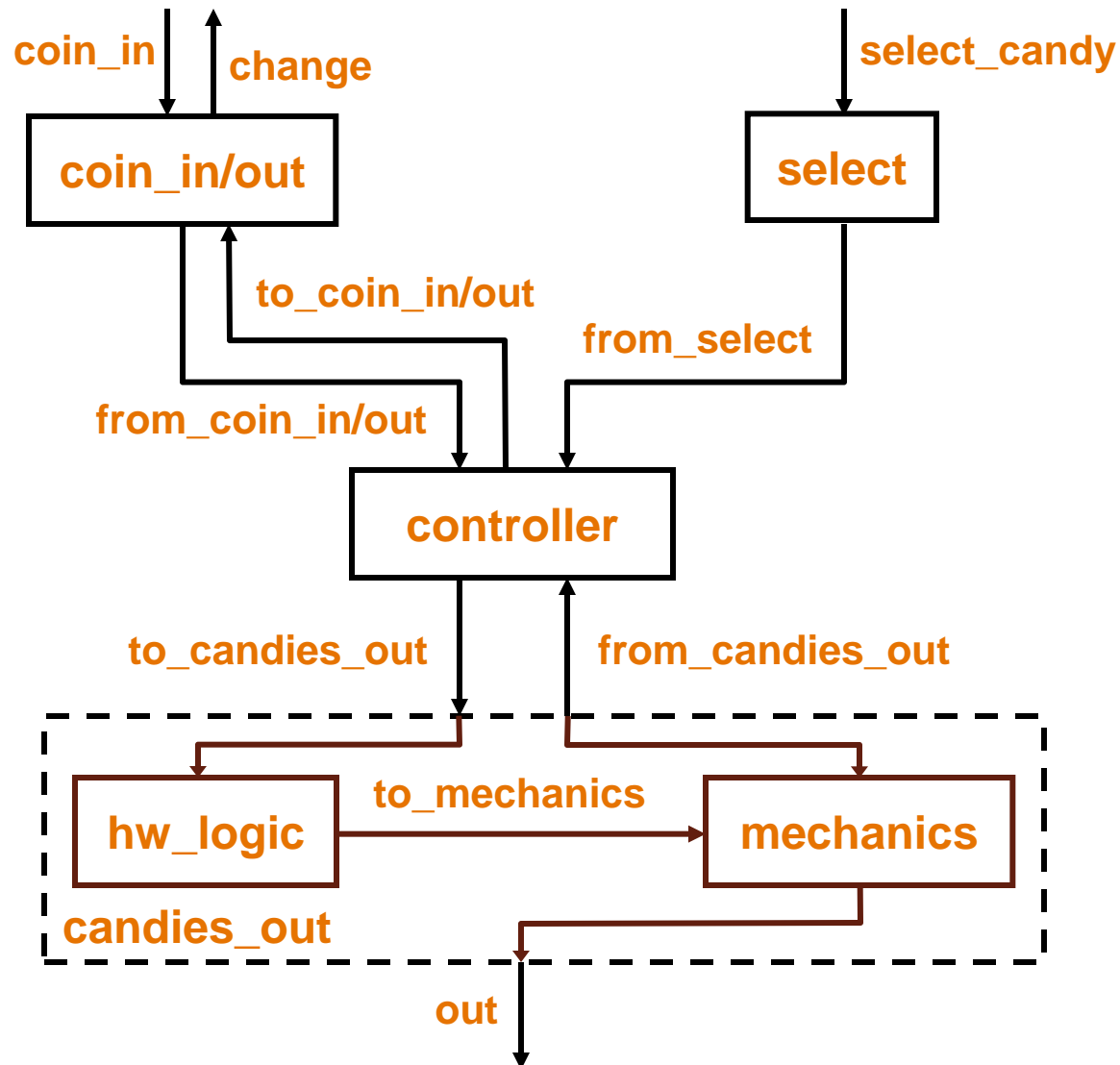
- $r_{n1}1 = \langle \text{good; in=a; good; out=a} \rangle$
- $r_{n1}2 = \langle \text{good; in=b; fty; out=b} \rangle$

- $r_{n2}1 = \langle \text{good; in=a; good; int=a} \rangle$
- $r_{n2}2 = \langle \text{good; in=b; fty; int=b} \rangle$
- $r_{n2}3 = \langle \text{fty; in=a; fty; out=a} \rangle$
- $r_{n2}4 = \langle \text{fty; in=b; fty; out=b} \rangle$

Édességautomata



Édességautomata finomítása



Finomítás ellenőrzés

1. Finomítási szabály-vezérelt tervezőrendszer
2. Definíciók alkalmazása próbálgatással
3. Véges automaták (FSM) felhasználása
 - Struktúra ellenőrzés
 - Csomópont-csomópont, illetve csomópont-részháló párok transzformációja → NDFST
 - Automata párokra biszimuláció keresése

Naplózott modellváltoztatások!

Adatfolyam csomópont → nemdeterminisztikus véges automata átalakítás

NDFST	jelentés	átírás az n csomópontból
Σ	Bemenő ABC	$(M_n)^i, i = I_n $
Γ	Kimenő ABC	$(M_n)^o, o = O_n $
S	Állapotok halmaza	S_n
S_0	Induló állapot	S_n^0
$\delta : S \times \Sigma \rightarrow S^*$	Állapotátmeneti függvény	$\delta \{ r_i(s), r_i(X_{in}), r_i(s') \}$
$\omega : S \times \Sigma \rightarrow \Gamma^*$	Kimeneti függvény	$\omega \{ (r_i(s), r_i(X_{in}), r_i(X_{out})) \}$

Adatfolyam hálózat →

nemdeterminisztikus véges automata átalakítás

DFN(N, C, S)

$$\Sigma = (M_n)^i, i = |I| \text{ és } n \in N$$

$$\Gamma = (M_n)^o, o = |O| \text{ és } n \in N$$

$$S = S_{DFN}, s_0 = s^0_{SFN}$$

$$\delta \{ s(fs), im(fs), s'(fs) \}, \forall fs \in FS$$

$$\omega \{ s(fs), im(fs), om(fs) \}, \forall fs \in FS$$

Véges automaták biszimulációja

Cél:

- eldönteni két véges automatáról, hogy ekvivalensen viselkednek-e?

Kibővített kimeneti függvény $\omega: S \times \Sigma^* \rightarrow \Gamma^*$

Definíció:

- ω_2 finomítása ω_1 -nek, ha $S_2 \supseteq S_1$ -nek, $\Sigma_2 \supseteq \Sigma_1$ -nek, Γ_2 pedig Γ_1 -nek finomítása

Definíció:

- $NDFST_1$ és $NDFST_2$ véges automaták között pontosan akkor van biszimuláció, ha ω_2 finomítása ω_1 -nek.

Finomítás ellenőrzése

Adatfolyam csomópont \rightarrow nemdeterminisztikus véges automata (NDFST₁)

Adatfolyam hálózat \rightarrow nemdeterminisztikus véges automata (NDFST₂)

- Akkor helyes a finomítás, ha NDFST₂ finomítása NDFST₁-nek

```
function Refinement_Check()
```

```
  if Diff_Struct() then return FAILURE
```

```
  for all  $n_1 \in N_1$ 
```

```
     $n_2$  vagy SDFN2 megfeleltetése  $n_1$ -nek
```

- n_1 transzformációja NDFST₁-vé
- n_2 vagy SDFN₂ transzformációja NDFST₂-vé

```
    if  $\neg$ Bisimulation() then return FAILURE
```

```
  end for
```

```
  return SUCCESS
```

```
end function
```

Modellbővítés

Modellezni kívánt mechanizmusok:

- Hibák
- Hibahatások
- Hibaterjedés

Ez megtehető a hibátlan modell szisztematikus kibővítésével (a hibamodell alapján).

Modellbővítés

1. Fizikai modell (alacsony szint)

- Szoros kapcsolat a fizikai hibaokkal

2. Logikai modell (magasabb szinteken)

- Modell perturbáció
 - Szisztematikus módon bővítik a modellt a hibás működéssel
 - if-then-else vagy switch-case típusú leírás
 - Perturbációk listája a hibalista
- Gráfmodellek
 - Csomópontok a rendszer komponenseit modellezik
 - Mindegyik leírásában egy előre kiszámított hibás működés
 - Hibás komponensnél hiba a kimeneten továbbterjed

Hibamodellezés

Neminterpretált (kvalitatív) modellezés esetén:

- Token lehet jó vagy hibás (színezés)
- Részletesebb hibamodell → többszínű

Hiba komolysága alapján:

- korrekt
- inkorrekt
- fatális
- katasztrofális

Lebegőpontos számítás eredménye:

- pontos
- közelítőleg pontos
- túl kicsi
- túl nagy

Hibamodellezés

Hibát jelző tokenek mellett a csomópontok állapotait is ki kell bővíteni hibát jelző állapotokkal.

→ Új tüzelési szabályok bevezetése

Hibatűrés aspektusai

error-free operation	$\langle \text{ok}; \text{in}=\text{ok}; \text{ok}; \text{out}=\text{ok}; 0 \rangle$
erroneous operation	$\langle \text{fty}; \text{in}=\text{ok}; \text{fty}; \text{out}=\text{fty}; 0 \rangle$
internal fault	$\langle \text{ok}; ; \text{fty}; ; 0 \rangle$
external fault	$\langle \text{ok}; \text{in}=\text{fty}; \text{fty}; \text{out}=\text{fty}; 0 \rangle$
repair	$\langle \text{fty}; \text{in}=\text{ok}; \text{ok}; \text{out}=\text{ok}; 0 \rangle$
error correction	$\langle \text{ok}; \text{in}=\text{fty}; \text{ok}; \text{out}=\text{ok}; 0 \rangle$
error masking	$\langle \text{fty}; \text{in}=\text{fty}; \text{fty}; \text{out}=\text{ok}; 0 \rangle$
error propagation	$\langle \text{ok}; \text{in}=\text{fty}; \text{ok}; \text{out}=\text{fty}; 0 \rangle$

Bizonytalanság modellezése

Szubjektív és nem teljes ismeretek ábrázolása.

Nemdeterminizmus lehet

- Funkcionális
- Alulspecifikáltság következménye

Bizonytalanság modellezése

Bizonytalanság

→ konkurens átmenetek

- különböző megoldások a választásra

- (Állapotátmeneti) valószínűségek

- sztochasztikus modellek (pl. Markov hálók)

- bonyolult kiértékelési módszerek

- Érték nemdeterminizmus

- küldött token értéke bizonytalan

- megoldható tüzelési (funkcionális) nemdeterminizmusra visszavezetve is

Édességautomata

komponens	hiba
pénz be/ki egység	ok rec
kiválasztó egység	ok cont
vezérlő egység	ok inc
édességkiadó egység	ok ctrl stk

Édességkiadó egység kibővített modellje

r1 = <ok; to_candies_out = ok; ok; from_candies_out = ok; out = ok; 0>

r2 = <ok; to_candies_out = inc; ok; from_candies_out = ok; out = inc; 0>

r3 = <ok; to_candies_out = x; ok; from_candies_out = ok; out = x; 0>

r4 = <ctrl; to_candies_out = ok; ctrl; from_candies_out = ok; out = inc; 0>

r5 = <ctrl; to_candies_out = inc; ctrl; from_candies_out = ok; out = x; 0>

r6 = <ctrl; to_candies_out = x; ctrl; from_candies_out = ok; out = x; 0>

r7 = <stk; to_candies_out = ok; stk; from_candies_out = dead; out = dead; 0>

r8 = <stk; to_candies_out = inc; stk; from_candies_out = dead; out = dead; 0>

r9 = <stk; to_candies_out = x; stk; from_candies_out = dead; out = dead; 0>