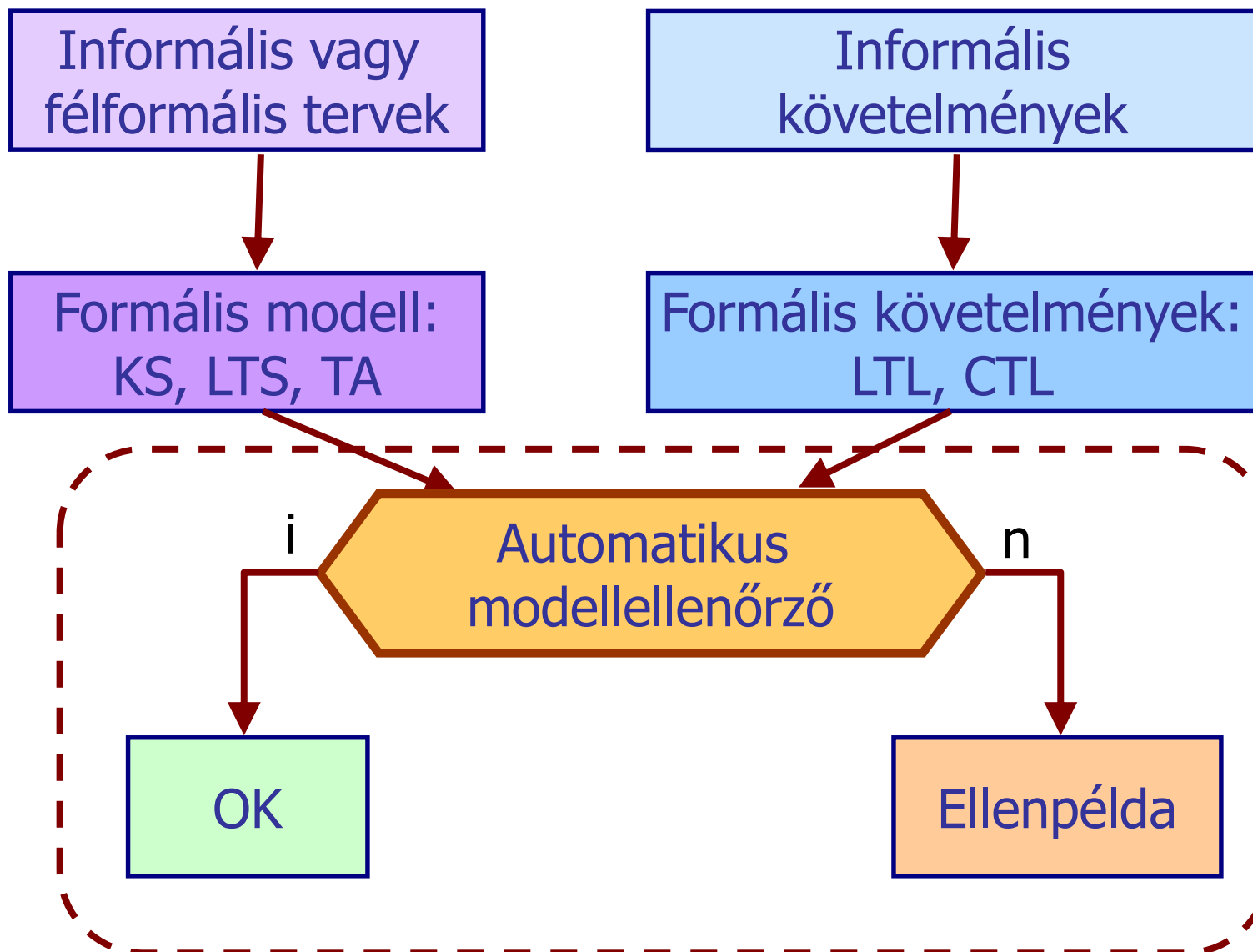


Modellellenőrző eszközök: A NuSMV eszköz (kedvcsináló)

dr. Majzik István

BME Méréstechnika és Információs Rendszerek Tanszék

Mire szolgálnak a modellellenőrzők?



Klasszikus eszközök

Eszköz	Modellek leírása	Tulajdonság leírása	Ajánlott használat
UPPAAL uppaal.org	Időzített automata, változókkal	Korlátozott CTL	Időfüggő viselkedés modellezése, szinkron kommunikáció
SPIN spinroot.com	Process Meta Language (Promela)	LTL, címkék, tulajdonság automata (never claim)	Aszinkron, üzenetekkel kommunikáló processzek protokolljai, algoritmusai
NuSMV nusmv.fbk.eu	Szinkron és aszinkron véges állapotú gépek (FSM)	CTL, LTL	Megosztott változókat használó komponensek algoritmusai, hardver rendszerek (szinkron)

A NuSMV modell leíró nyelve (1)

- Véges állapotú gép (FSM)
 - „Lehetséges következő állapot” reláció definiálása az állapotok között
 - Változók használhatók típussal: `boolean`, `integer`, `enum`, `array`
- Változók deklarálása
 - `VAR` szekció a modellben: `identifier : type;`
- A modell kezdőállapota: Kezdeti értékadás
 - `ASSIGN` szekcióban: `init(identifier) := simple_expression;`
 - Értékadás nélküli változó: `bemenet` (bármilyen lehet, típusának megfelelően)
- A modell állapotátmenete: Változók értékének változása
 - `ASSIGN` szekcióban: `next(identifier) := next_expression;`
a kifejezésben az aktuális és következő állapot változói is szerepelhetnek (utóbbi a `next()` operátorral)
 - `ASSIGN` szekcióban: `identifier := simple_expression;`
megadja a változó értékét minden állapotra

A NuSMV modell leíró nyelve (2)

- Feltételes kifejezések

- if-then-else kifejezés a szokásos (C-szerű) szintaxis szerint határozza meg a változók új értékét

`condition ? expression1: expression2`

- case kifejezés: az első olyan ág, aminek feltétele igaz, határozza meg a változók új értékét (hibajelzés, ha nincs igaz feltétel vagy TRUE ág)

`case`

`condition1 : expression1;`

`...`

`conditionn : expressionn;`

`TRUE: expressiondefault;`

`esac`

- Változó értékadás kényszerekkel (logikai kifejezés)

- **INIT** szekcióban: Kezdőérték olyan lehet, ami a kényszert kielégíti
- **TRANS** szekcióban: Aktuális és új értékek (ld. **next()** operátor) ki kell elégítsék a kényszert

Példa modell: Producer

```
MODULE main
```

```
  VAR
```

```
    request: boolean;
```

```
    state: {ready, busy};
```

```
  ASSIGN
```

```
    init(state) := ready;
```

```
    next(state) :=
```

```
      case
```

```
        state = ready & request: busy;
```

```
        state = busy:           {ready, busy};
```

```
        TRUE: ready;
```

```
      esac;
```

A NuSMV modell leíró nyelve (3)

„Hagyományos” vezérlési struktúrák

- if() feltétel

if ($x < S$ & $b > 0$)

next(x) := $x + 1$

- for(; ;) ciklus

for ($j = 1$; $j \leq N - 1$; $j = j + 1$)

next($a[j]$:= $a[j - 1]$)

A NuSMV tulajdonság leíró nyelve

- Invariánsok
 - **INVAR** szekcióban logikai kifejezés a változók értékére
- LTL kifejezések
 - **LTLSPEC** szekció, standard jelölés
 - Atomi kijelentések helyett logikai kifejezések
 - Pl.: **LTLSPEC** $G (y=4 \rightarrow X (y=6))$
- CTL kifejezések
 - **CTLSPEC** vagy **SPEC** szekció, standard jelölés
 - Atomi kijelentések helyett logikai kifejezések
 - Pl.: **CTLSPEC** $AG(request \rightarrow AF(state = busy))$
- Hasznos: Kifejezések helyettesítése (makró)
 - **DEFINE** szekcióban: $alias := simple_expression$

Moduláris felépítés

- Alapegység:
 - **MODULE** névvel ellátva, paramétere is lehet
 - Pl. **MODULE** user(semaphore)
- Modulok példányosítása a VAR szekcióban
 - Simán (paraméterezéssel): szinkron komponens
Pl.: **proc1** : user(semaphore);
 - **process** kulcsszóval: aszinkron komponens
Pl.: **proc1** : process user(semaphore);
- Fair viselkedés megadása a FAIRNESS szekcióban
 - **running** kulcsszó: végtelenül sokszor fut a processz
 - CTL állapotkifejezés: végtelenül sokszor igaz lesz

Szemantika: Szinkron vagy aszinkron

- Szinkron végrehajtás
 - Modulok sima példányosítása
 - Egy „lépésben” mindegyik modul egy-egy átmenetet hajt végre (új változó értékek kiszámításával)
 - Elsősorban hardver ellenőrzéshez (\sim órajelre működik)
- Aszinkron végrehajtás
 - Modulok process kulcsszóval történő példányosítása a fő modulban (processzek)
 - Egy „lépésben” egy véletlenszerűen kiválasztott modul egy átmenetet hajt végre (új változó értékek kiszámítása)
 - Tipikusan elosztott rendszerek ellenőrzésére, amelyek megosztott változókat használnak

Szinkron illetve aszinkron rendszer

```
MODULE cell(input)
```

```
  VAR
```

```
    val : {red, green, blue};
```

```
  ASSIGN
```

```
    next(val) := {input};
```

```
MODULE main
```

```
  VAR
```

```
    c1 : cell(c3.val);
```

```
    c2 : cell(c1.val);
```

```
    c3 : cell(c2.val);
```

```
MODULE cell(input)
```

```
  VAR
```

```
    val : {red, green, blue};
```

```
  ASSIGN
```

```
    next(val) := {input};
```

```
MODULE main
```

```
  VAR
```

```
    c1 : process cell(c3.val);
```

```
    c2 : process cell(c1.val);
```

```
    c3 : process cell(c2.val);
```

Példa aszinkron rendszerre

```
MODULE main
```

```
VAR
```

```
semaphore : boolean;
```

```
proc1 : process user(semaphore);
```

```
proc2 : process user(semaphore);
```

```
ASSIGN
```

```
init(semaphore) := FALSE;
```

```
CTLSPEC AG ! (proc1.state = critical &  
              proc2.state = critical)
```

```
CTLSPEC AG (proc1.state = entering ->  
            AF proc1.state = critical)
```

```
LTLSPEC G ! (proc1.state = critical &  
              proc2.state = critical)
```

```
LTLSPEC G (proc1.state = entering ->  
            F proc1.state = critical)
```

```
MODULE user(semaphore)
```

```
VAR
```

```
state : {idle, entering, critical, exiting};
```

```
ASSIGN
```

```
init(state) := idle;
```

```
next(state) :=
```

```
case
```

```
state = idle           : {idle, entering};
```

```
state = entering & !semaphore : critical;
```

```
state = critical       : {critical, exiting};
```

```
state = exiting        : idle;
```

```
TRUE : state;
```

```
esac;
```

```
next(semaphore) :=
```

```
case
```

```
state = entering : TRUE;
```

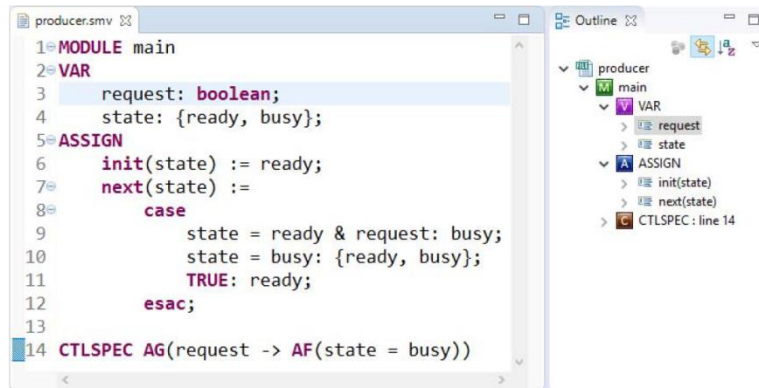
```
state = exiting  : FALSE;
```

```
TRUE : semaphore;
```

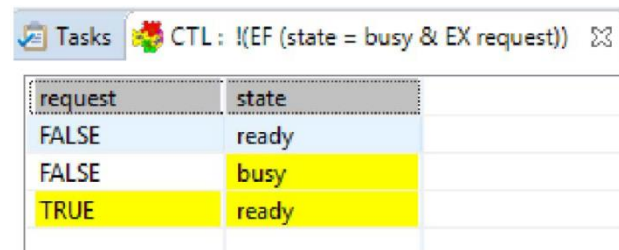
```
esac;
```

A NuSMV modellellenőrző

- Parancssori verzió
 - Végrehajtás: nusmv model
 - Szöveges kimenet
 - Ellenpélda is szövegesen (lépések, változók értékei)
- Eclipse keretben: NuSeen
 - Xtext alapú modell szerkesztő
 - Táblázatos ellenpélda megjelenítés
 - Változók függőségi gráfja



```
1=MODULE main
2=VAR
3=  request: boolean;
4=  state: {ready, busy};
5=ASSIGN
6=  init(state) := ready;
7=  next(state) :=
8=    case
9=      state = ready & request: busy;
10=     state = busy: {ready, busy};
11=     TRUE: ready;
12=    esac;
13=
14=CTLSPEC AG(request -> AF(state = busy))
```



Tasks CTL : !(EF (state = busy & EX request))

request	state
FALSE	ready
FALSE	busy
TRUE	ready