

# Részletes tervek ellenőrzése

Majzik István

<http://www.inf.mit.bme.hu/>

1

## Tartalomjegyzék

- **Áttekintés**
  - Milyen szerepe van a részletes terveknek?
  - Milyen ellenőrzési módszerek vannak?
- **Modellellenőrzés**
  - Kimerítő (teljes) technikák
  - Korlátos modellellenőrzés
  - UML állapottérképek modellellenőrzése
- **Ekvivalencia ellenőrzés**
  - Trace ekvivalencia
  - Megfigyelési ekvivalencia (gyenge biszimuláció)
  - Egy rendezési reláció a lehetséges viselkedés szerint
  - Mintapélda: Hibatúrás verifikációja

2

# Részletes tervezés

- A részletes tervezés szerepe
  - Rendszerszintű részletes tervezés
    - Rendszerszintű algoritmusok: Modulok együttműködése
    - Globális adatstruktúrák: Több modul által használtak
  - Komponens (modul) szintű részletes tervezés
    - Belső algoritmusok
    - Belső adatstruktúrák
- Módszerek jellegzetességei
  - Viselkedés leírása hangsúlyos: Állapotok, akciók, időbeliség, ...
  - Adatstruktúrák leírása: Absztrakt adatstruktúrák → implementáció
- Tervezési, modellezési nyelvek
  - Skálázható: Formális, félformális, strukturált módszerek
  - Modul szinten: Implementáció-közeli „pseudo-kód” is lehet
- Verifikációs szempontok
  - Megfelelőség a szoftverkövetelmény-specifikációnak
  - Megfelelőség előző fejlesztési lépések döntéseinek

3

# Ellenőrzési módszerek

- Statikus ellenőrzés: Felülvizsgálat, egyenrangú átvizsgálás
  - Ellenőrzőlisták, hibabecslés
    - Teljesség, ellentmondásmentesség, megvalósíthatóság, tesztelhetőség
  - Követhetőség felhasználása
  - Vezérlési folyamat elemzés
    - Strukturáltság
    - Határérték elemzés
  - Adatáramlás elemzés
    - Értékadás és felhasználás kapcsolata
  - „Alattomos komponensek” elemzése
    - Nemkívánatos egymásrahatás vagy információáramlás
  - Szimbolikus végrehajtás
- Dinamikus ellenőrzés:
  - Szimuláció (modellek alapján)
  - Prototípus készítés és animáció
- Formális verifikáció

4

# Formális verifikáció

- Matematikai eszközök használata (nyelv + algoritmus)
  - Formális nyelv: Formális szintaxis és szemantika
    - Viselkedés leírás (terv, implementáció)
    - Tulajdonság leírás (követelmények, specifikáció)
- A verifikáció módja
  - „Önmagában való” vizsgálat
    - Pl. tervek, modellek ellentmondás-mentessége
  - „Elvárt tulajdonságok” ellenőrzése
    - Pl. szoftverkövetelmény-specifikáció betartása
  - „Változások” vizsgálata
    - Pl. tervezői döntések hatása: tulajdonság megtartása, funkciók lefedése
- Ideális formális verifikáció
  - Absztrakció feltételezéseinek ellenőrzése (környezet, prototípus)
  - Verifikációs eszközök korrektségének belátása

5

## Tervek és modellek szerepe a formális verifikációban

- Viselkedés leírás
  - Alapszintű:
    - KS, LTS, KTS, automaták, Büchi automaták
  - Magasabb szintű:
    - Vezérlés orientált: Hierarchikus automata, Petri-háló
    - Adatfeldolgozás orientált: Adatfolyam háló
    - Kommunikáció orientált: Processz algebrák
  - Mérnöki modellek:
    - UML diagramok formális szemantikával
- Tulajdonság leírás
  - Alapszintű:
    - Elsőrendű logika, temporális logika, referencia automata
  - Magasabb szintű:
    - MSC, LSC, ...

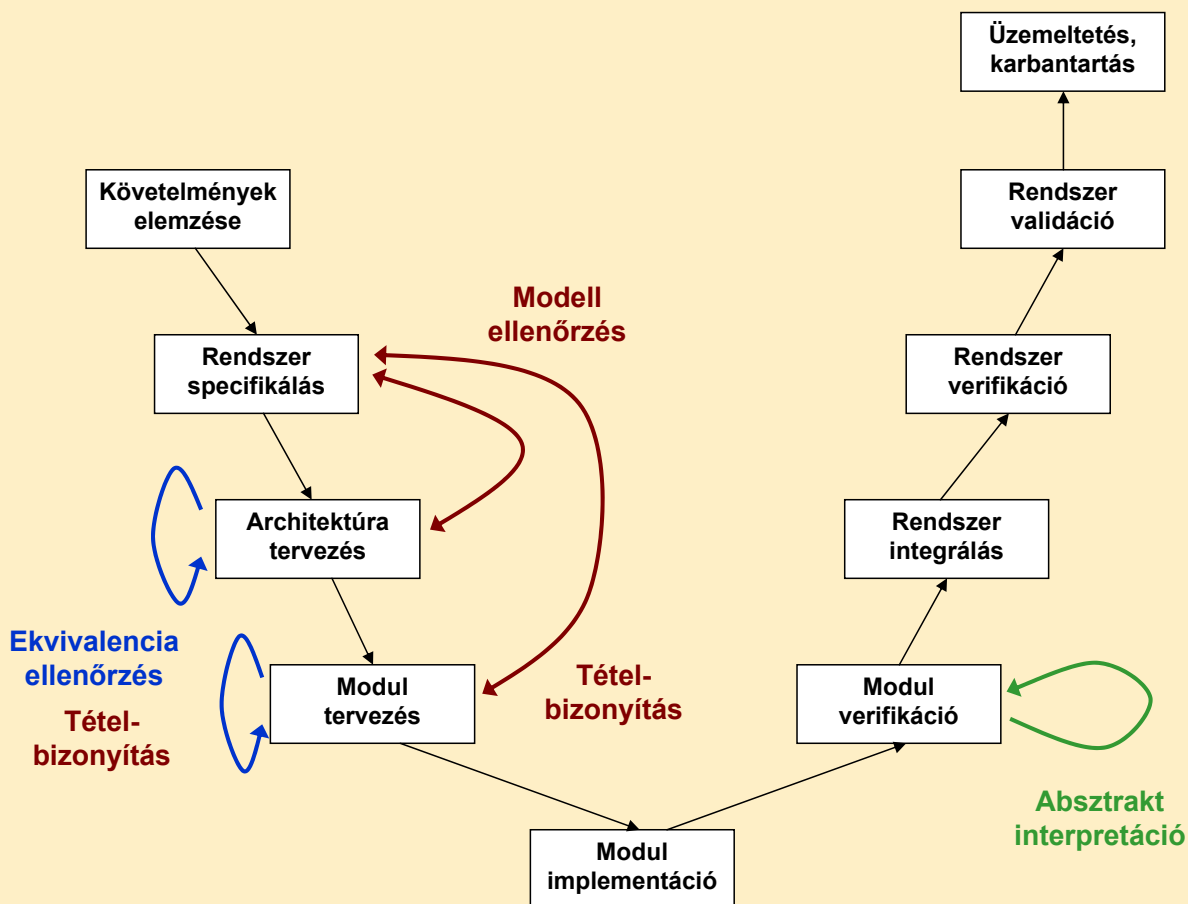
6

# A legelterjedtebb formális verifikációs technikák

Modell / technika	Viselkedés modell (alapszintű)	Tulajdonság modell (alapszintű)
<b>Modellellenőrzés</b>	Kripke-struktúra	Temporális logika (elsőrendű logika)
<b>Ekvivalencia ellenőrzés</b>	LTS (Labeled Transition System), automata	LTS, automata (referencia viselkedés)
<b>Tételbizonyítás</b>	Dedukciós rendszer	Elsőrendű logika (bizonyítandó tétel)
<b>Statikus analízis (absztrakt interpretáció)</b>	Kripke-struktúra (programból absztrakcióval)	Elsőrendű logika, assertion

7

# A formális verifikációs technikák szerepe



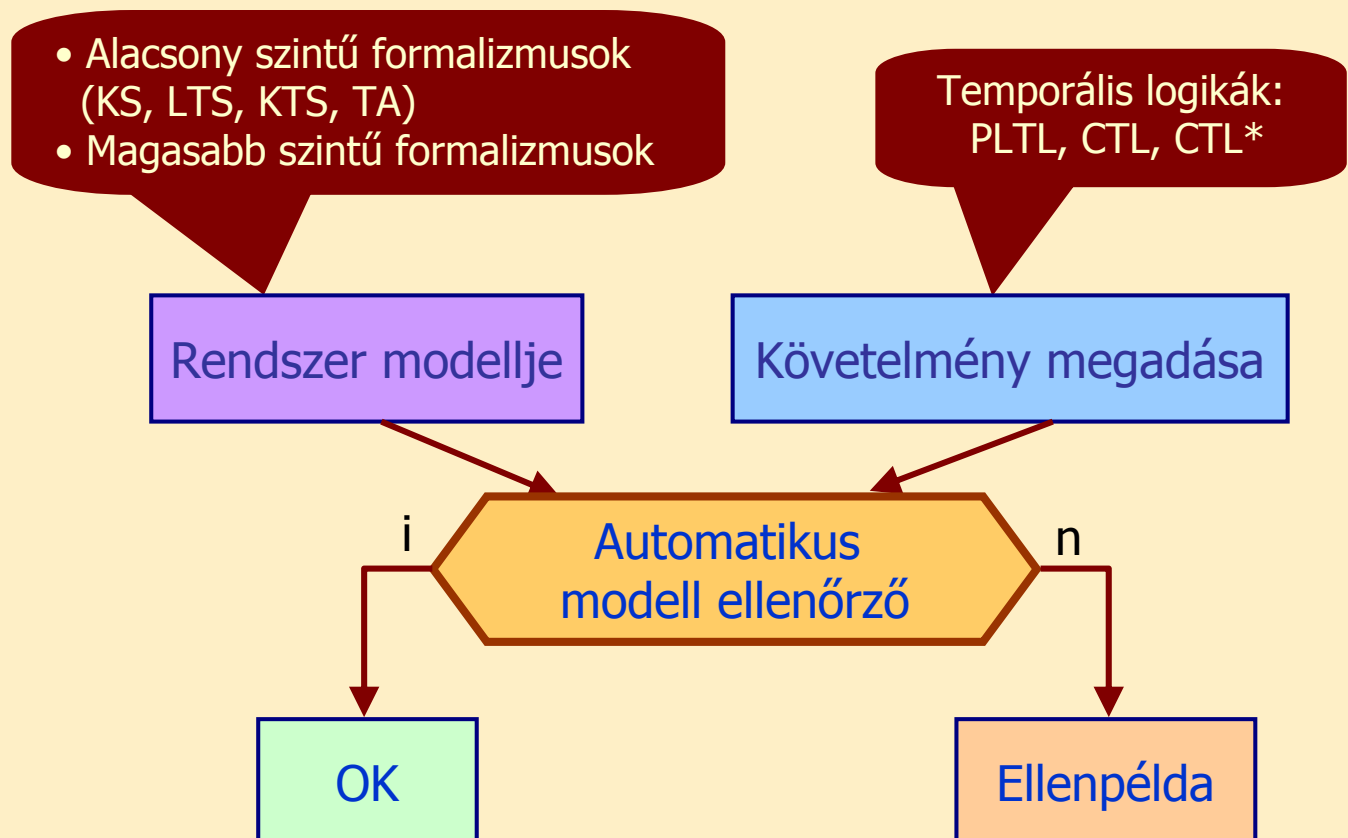
8

# Tartalomjegyzék

- **Áttekintés**
  - Milyen szerepe van a részletes terveknek?
  - Milyen ellenőrzési módszerek vannak?
- **Modellellenőrzés**
  - Teljes (kimerítő) technikák
  - Korlátos modellellenőrzés
  - UML állapottérképek modellellenőrzése
- **Ekvivalencia ellenőrzés**
  - Trace ekvivalencia
  - Megfigyelési ekvivalencia (gyenge biszimuláció)
  - Egy rendezési reláció a lehetséges viselkedés szerint
  - Mintapélda: Hibatúrés verifikációja

9

## Modellellenőrzés



10

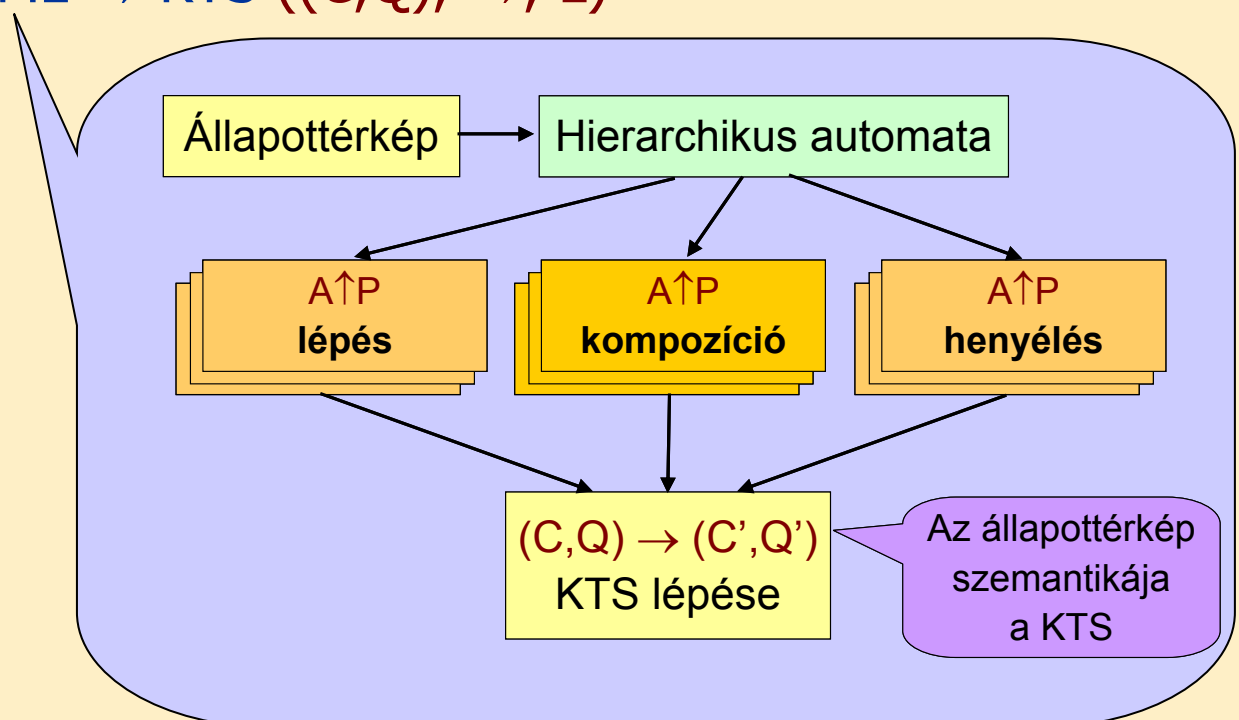
# Modellellenőrzési technikák

- Hátér algoritmusok (ld. Formális módszerek)
  - Teljes modellellenőrzés
    - Tabló alapú (kibontás)
    - Szemantikán alapuló „fixpont” algoritmus (állapottér bejárás)
    - Hatékony állapottér reprezentáció (BDD)
  - Korlátozott modellellenőrzés
    - Az állapottérben adott mélységig ellenőriz (kezdőállapottól indulva adott végrehajtási útvonal hosszig)
    - Gyorsan fejlődő SAT technikákra visszavezethető
- Előnyök és hátrányok
  - ☺ Teljesen automatikus ellenőrzés
  - ☺ Ellenpélda generálás (hibakereséshez)
  - ☹ Vezérlés-orientált ellenőrzés
  - ☹ Állapottér robbanás (részben kezelhető)

11

## Példa: UML állapottérképek modellellenőrzése

- Viselkedés-megtartó modelltranszformáció:  
UML  $\rightarrow$  KTS  $((C,Q), \rightarrow, L)$



12

## Példa: UML állapottérképek modellellenőrzése

- Viselkedés-megtartó modelltranszformáció:  
UML → Timed Automata (UPPAAL)

- Aktív objektumok (osztályok) leképezése automatákká
- Eseménykezelés leképezése
  - Külső környezet modellezése: Eseményeket „generál”
  - Eseménysor, esemény ütemező, RTC lépések
- Adatkezelés leképezése
  - Egyszerű adattípusok, korlátozott adattartományok
- Konkurens és hierarchikus állapotmodell leképezése
  - Állapotkonfigurációk
- Időkezelés (kiterjesztés)
  - Megfigyelő (clock observer) idő eseményeket generál

13

## Tartalomjegyzék

- **Áttekintés**
  - Milyen szerepe van a részletes terveknek?
  - Milyen ellenőrzési módszerek vannak?
- **Modellellenőrzés**
  - Kimerítő (teljes) technikák
  - Korlátos modellellenőrzés
  - UML állapottérképek modellellenőrzése
- **Ekvivalencia ellenőrzés**
  - Trace ekvivalencia
  - Megfigyelési ekvivalencia (gyenge biszimuláció)
  - Egy rendezési reláció a lehetséges viselkedés szerint
  - Mintapélda: Hibatúrás verifikációja

14

# Használat: Tervezői döntések ellenőrzése

- **Megfelelőség (ekvivalencia) modellek között**  
Specifikáció (absztrakt)  $\leftrightarrow$  Megvalósítás (részletes)  
Elvárt protokoll  $\leftrightarrow$  Kínált protokoll  
Ideális rendszer  $\leftrightarrow$  Hibatűrő rendszer hibák mellett
- **Finomítás (rendezés) modellek között:**
  - Nemdeterminizmus csökkenése
  - Viselkedés bővülése (eredeti viselkedés megtartása)

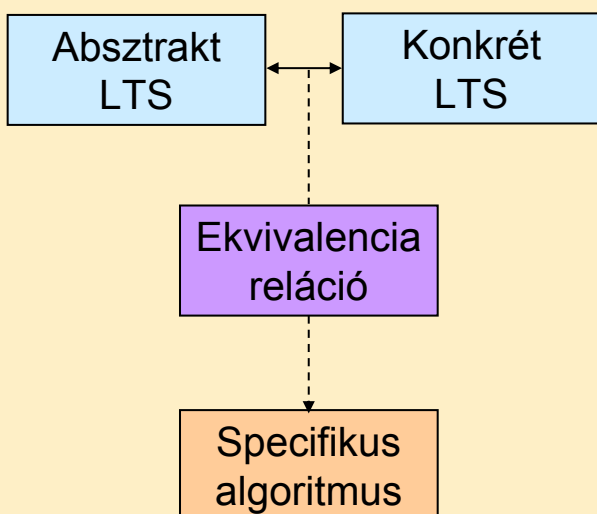
## Használt relációk:

- **Ekvivalencia:** Reflexív, tranzitív, szimmetrikus
- **Rendezés:** Reflexív, tranzitív, antiszimmetrikus

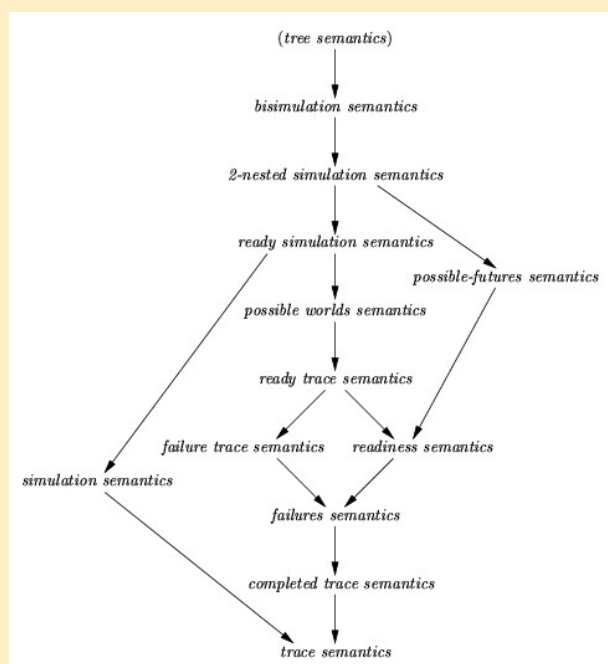
15

## A probléma formalizálása

### Ekvivalencia ellenőrzés:



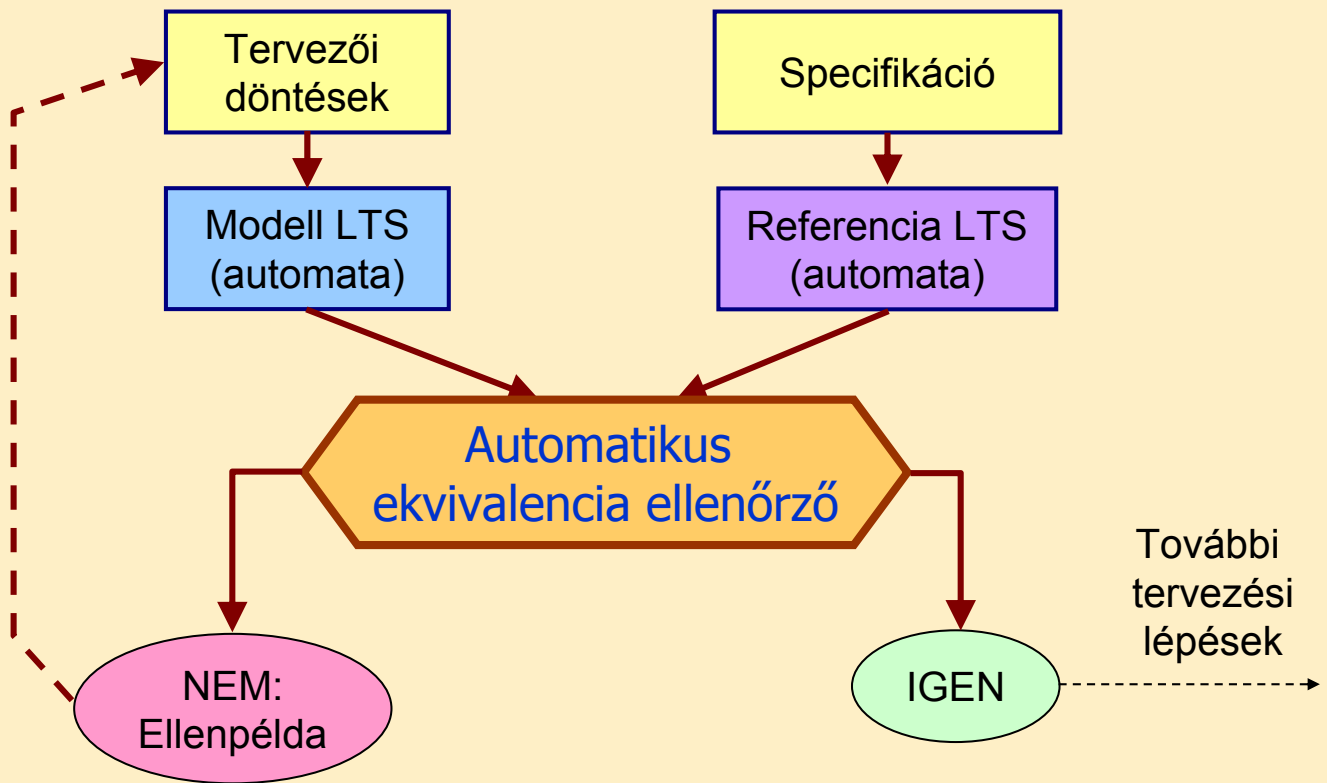
### Ekvivalencia relációk:



16



## Ekvivalencia ellenőrzés



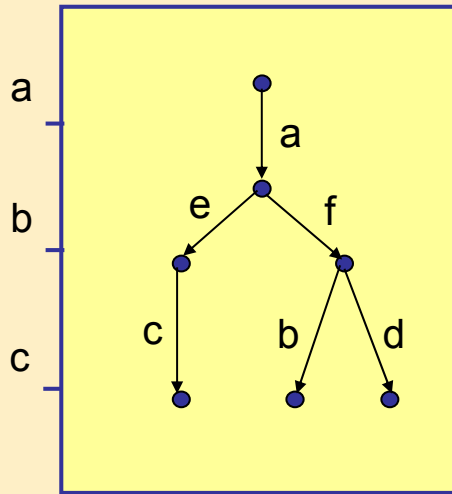
17

## A komponensek „fekete doboz” modellje

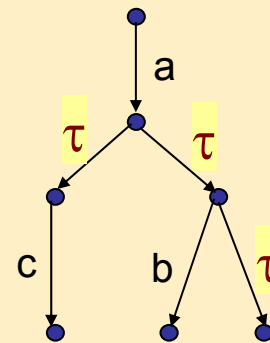
- **Megfigyelhető akciók**
  - A vizsgált komponens (modul) interfészen megjelenő, a környezet számára érdekes („releváns”) viselkedés
    - Metódus hívása, metódushívás fogadása
    - Üzenet küldése, üzenet fogadása
- **Nem megfigyelhető akciók ( $i, \tau$ )**
  - Az interfészen nem megjelenő, vagy a környezet számára nem érdekes („don't care”) viselkedés
    - Belső működés (hívások, üzenetek)
    - Figyelmen kívül hagyható hívások, üzenetek
- **Nemdeterminizmus**
  - A környezet szempontjából nem érdekes belső működés is okozhatja (ezt eltakarja a fekete doboz modell)

18

# Megfigyelhető viselkedés



Komponens  
belső viselkedés



Megfigyelhető  
viselkedés

19

## Trace ekvivalencia: Jelölések

- Minta: Automaták

$$A_1 = A_2 \text{ ha } L(A_1) = L(A_2)$$

- LTS-ek esetén:

- Minden állapot elfogadó állapot
- „Nyelv”: Minden akciószekvencia (trace)

- Jelölések:

$\alpha = a_1 a_2 a_3 a_4 \dots a_n \in Act^*$  véges akciószekvencia ( $\varepsilon$  az üres)

$s \xrightarrow{\alpha} s'$  ha  $\exists s_0 s_1 \dots s_n$  állapotsorozat ahol  $s_0 = s$ ,  $s_n = s'$ ,  $s_i \xrightarrow{a_{i+1}} s_{i+1}$

$\Lambda(s)$  legyen  $s$  trace-einek halmaza:  $\Lambda(s) = \left\{ \alpha \mid \exists s' : s \xrightarrow{\alpha} s' \right\}$

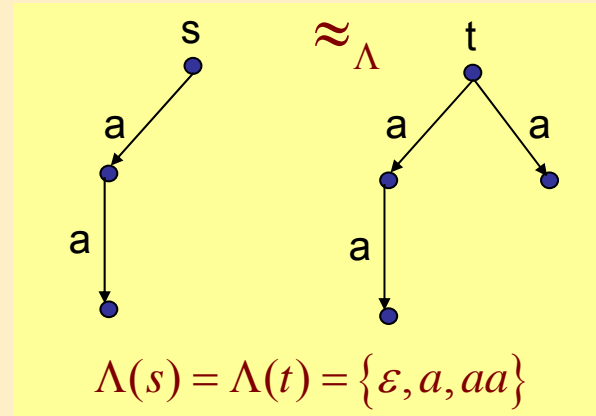
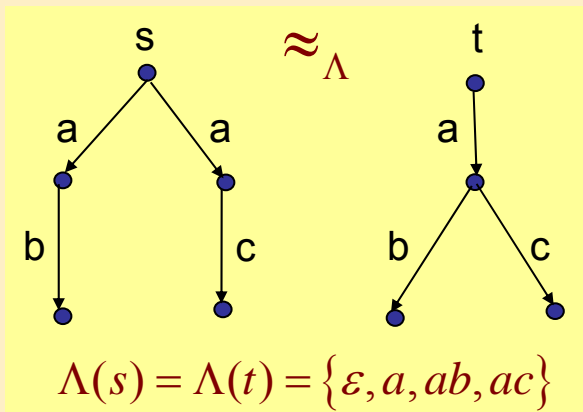
20

## Trace ekvivalencia: Definíció és példák

- Legyen  $T_1$  és  $T_2$  két LTS,  $s_1$  és  $s_2$  kezdőállapottal
- Definíció:

$$T_1 \approx_{\Lambda} T_2 \text{ a.cs.a. } \Lambda(s_1) = \Lambda(s_2)$$

- Példák:



- Problémák:

- Ekvivalens LTS-ek eltérő viselkedése pl. megállás szempontjából
- Azonos trace-ek különböző állapotokon keresztül

21

## Megfigyelési ekvivalencia (gyenge biszimuláció)

- Jellegzetességek

- Lényege: Két megfigyelhető viselkedés ekvivalens, ha egymást szimulálni tudják:
  - Azonosak a megfigyelhető akciószekvenciák
  - Ekvivalens állapotokon keresztül figyelhetők meg
- Nem érzékeny a hatás nélküli belső átmenetekre (ezek megjelenésére, számára)

- Jelölések:

$\alpha \in Act^*$  véges akciószekvencia ( $\varepsilon$  az üres)

$\hat{\alpha} \in (Act - \tau)^*$  megfigyelhető akciószekvencia ( $\tau$  törlésével)

itt  $\hat{\alpha} = \varepsilon$  ha  $\alpha = \tau$

$s \xRightarrow{\beta} s'$  ha  $\exists \alpha: s \xrightarrow{\alpha} s'$  és  $\beta = \hat{\alpha}$

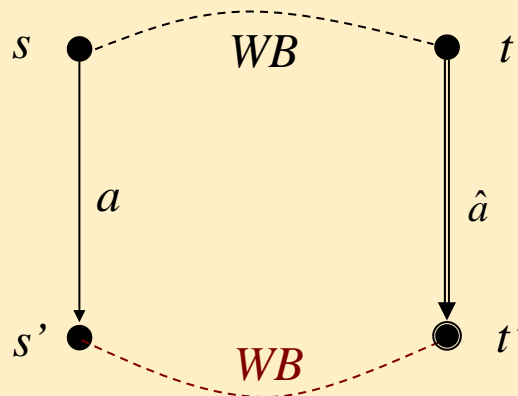
22

# Gyenge biszimuláció: Definíció

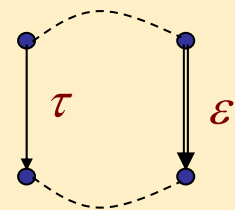
- Definíció:

$WB \subseteq S \times S$  gyenge biszimuláció, ha minden  $(s, t) \in WB$  és bármely  $a \in Act$ ,  $s', t' \in S$  esetén fennáll:

- ha  $s \xrightarrow{a} s'$  akkor  $\exists t' : t \xRightarrow{\hat{a}} t'$  és  $(s', t') \in WB$
- ha  $t \xrightarrow{a} t'$  akkor  $\exists s' : s \xRightarrow{\hat{a}} s'$  és  $(s', t') \in WB$



Extrém eset:



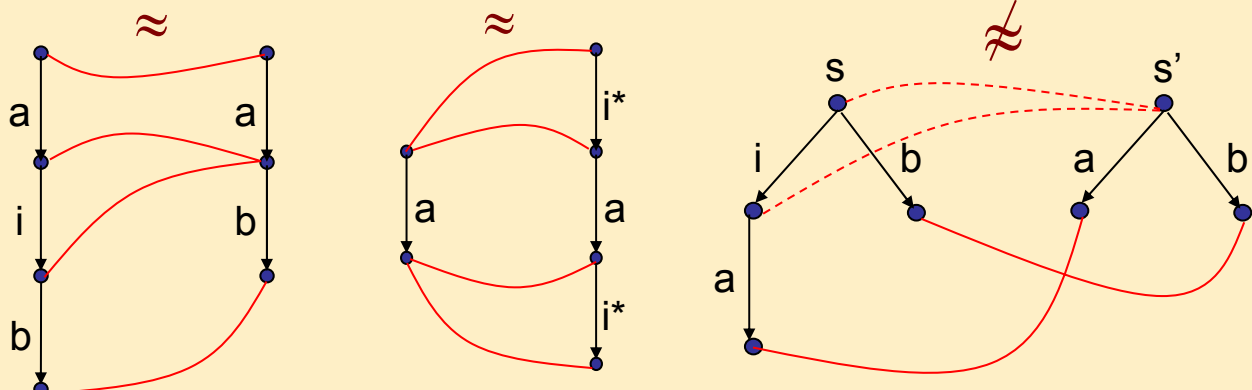
# Megfigyelési ekvivalencia: Definíció és példák

- Megfigyelési ekvivalencia

= gyenge biszimuláció ekvivalencia (observation equivalence)

$$T_1 \approx T_2 \text{ a.cs.a. } s_1 \approx s_2 \text{ azaz } \exists WB : (s_1, s_2) \in WB$$

- Példák



# Ekvivalencia relációk számítási módszere

## Partíció finomítás:

1. Kezdetben minden állapotpár eleme a relációnak  
Egy partíciót (ekvivalencia osztályt) képeznek
2. Minden állapotpárra:  
Ha az egyikből indulva van olyan átmenet, amit a másik nem tud a definíció szerint szimulálni, akkor
  - Az adott állapotpár kizárása (nem ekvivalensek)
  - Következmények végigvezetése a bejövő átmenetek végén lévő állapotokra
    - Nem ekvivalensek, ha nem ekvivalens állapotokba kerülnek
3. Ha már nincs változás (fixpont):  
Végleges ekvivalencia osztályok adódtak  
Ha a kezdőállapotok azonos ekvivalencia osztályban vannak, akkor az LTS-ek ekvivalensek

25

## Egy rendezési reláció: Lehetséges viselkedés szerint

### • Jelölések:

$\beta \in (Act - \tau)^*$  megfigyelhető akciószekvencia ( $\tau$  törlése)

$s \xRightarrow{\beta} s'$  ha  $\exists \alpha \in Act^*: s \xrightarrow{\alpha} s'$  és  $\beta = \hat{\alpha}$

$\Delta(s)$  az  $s$ -ből induló megfigyelhető akciószekvenciák halmaza:

$$\Delta(s) = \left\{ \beta \mid \exists s' : s \xRightarrow{\beta} s' \right\}$$

### • Lehetséges viselkedés szerinti rendezés:

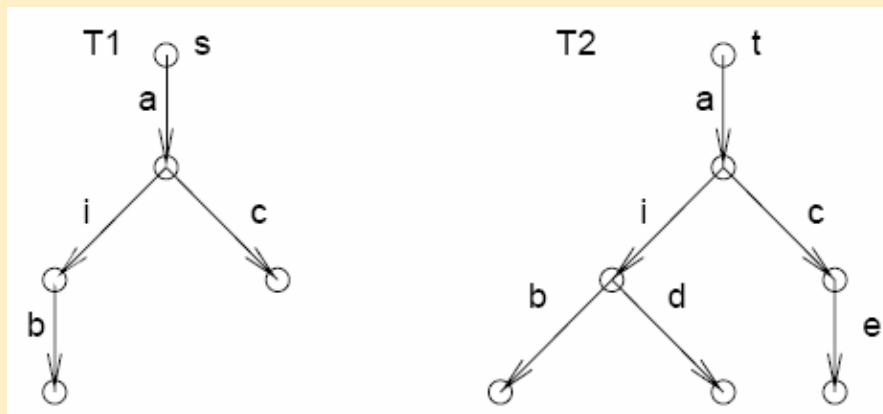
$$T_1 \leq_{\Delta} T_2 \text{ a.cs.a. } \Delta(s_1) \subseteq \Delta(s_2)$$

itt  $T_2$ -ben több a megfigyelhető akciószekvencia

26

# Lehetséges viselkedés szerinti rendezés: Példa

- Példa:



$$\Delta(s) = \{a, ab, ac\}$$

$$\Delta(t) = \{a, ab, ac, ad, ace\}$$

27

## Lehetséges viselkedés szerinti rendezés és a tesztelés

- **Kapcsolat a teszteléssel:**  $T_1 \leq_{\Delta} T_2$  esetén:
  - Minden teszt, ami  $T_1$  esetén sikeres lehet,  $T_2$  esetén is sikeres lehet, de belső akciók miatt sikertelen is lehet (nemdeterminizmus)
  - Másképp:  $T_1$  lehetséges sikeres tesztjei  $T_2$  lehetséges sikeres tesztjei között vannak
- **Jellegzetességek:**
  - Olyan finomítást definiál, amelynek során **nem „veszik el”** lehetséges megfigyelhető viselkedés
- **Egy másik rendezési reláció:**
  - Olyan finomítás, ami a **mindig sikeres** tesztekre („kötelező” viselkedésre) ad megkötést

28

# Tartalomjegyzék

- **Áttekintés**
  - Milyen szerepe van a részletes terveknek?
  - Milyen ellenőrzési módszerek vannak?
- **Modellellenőrzés**
  - Kimerítő (teljes) technikák
  - Korlátos modellellenőrzés
  - UML állapotterképek modellellenőrzése
- **Ekvivalencia ellenőrzés**
  - Trace ekvivalencia
  - Megfigyelési ekvivalencia (gyenge biszimuláció)
  - Egy rendezési reláció a lehetséges viselkedés szerint
  - **Mintapélda: Hibatúrés verifikációja**