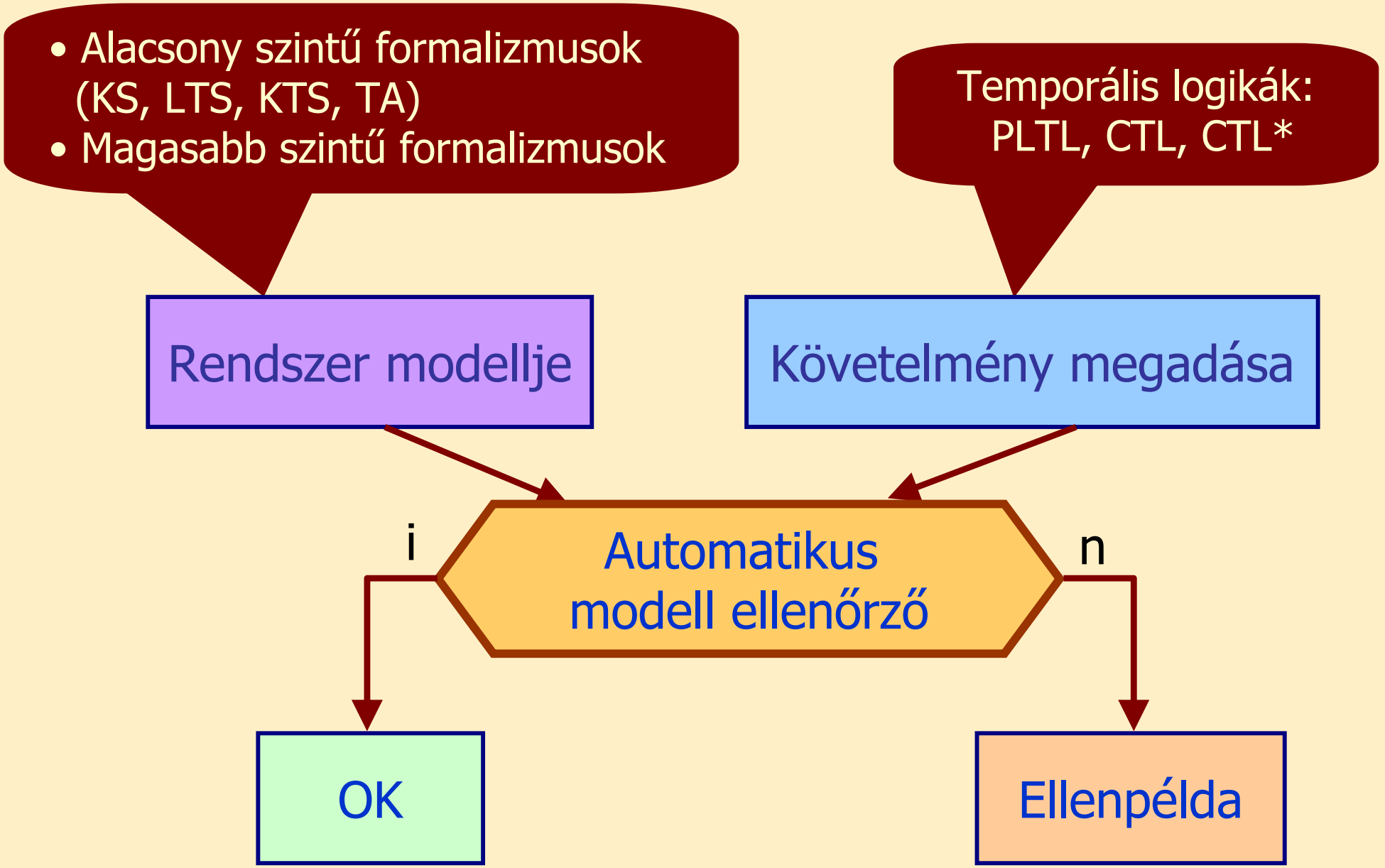
Modellellenőrzés

dr. Majzik István
BME Méréstechnika és Információs Rendszerek Tanszék

Motiváció: Mit szeretnénk elérni?



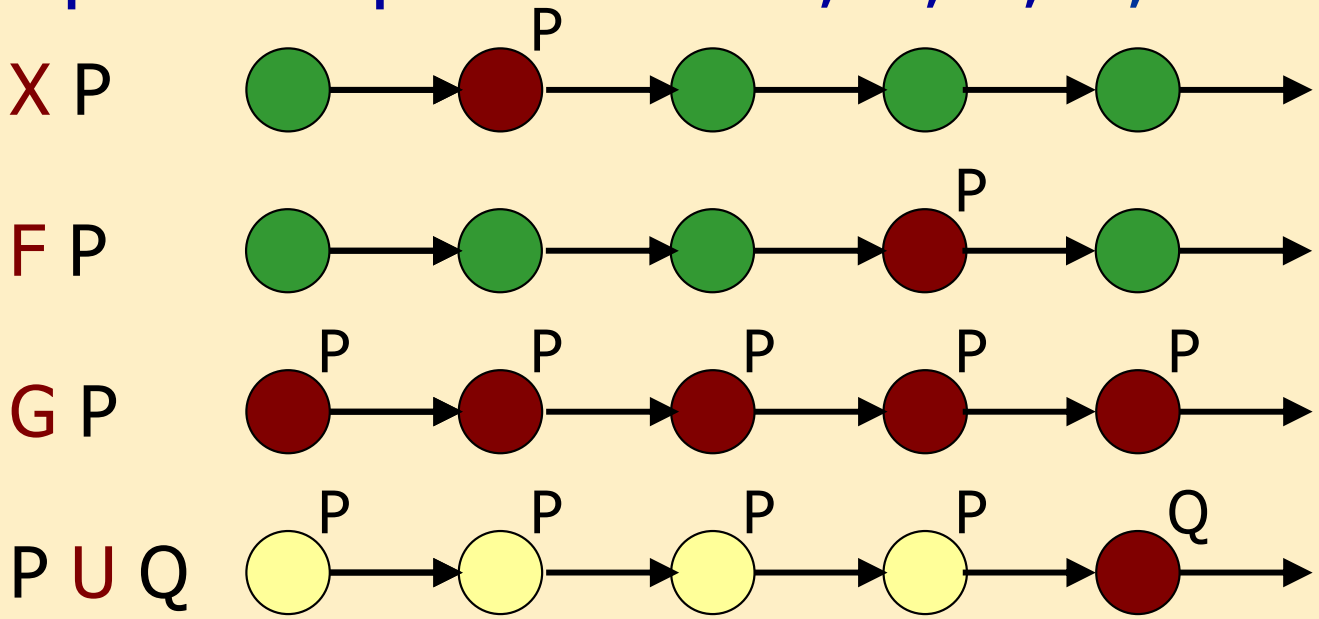
Ismétlés: Lineáris idejű temporális logika: PLTL

PLTL (Propositional Linear Time Temporal Logic)

p, q, r, ... kifejezések konstruálása:

- Atomi kijelentések (AP elemei): P, Q, ...
- Boole logikai operátorok: $\wedge$, $\vee$, $\neg$, $\Rightarrow$

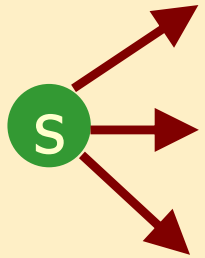
  $\wedge$: És, $\vee$: Vagy, $\neg$: Negálás , $\Rightarrow$: Implikáció
- Temporális operátorok: F, G, X, U, informálisan:



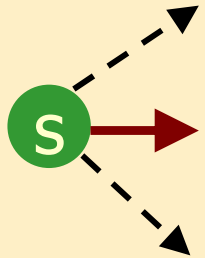
Ismétlés: Elágazó idejű temporális logika: CTL*

- Útvonalakon értelmezett operátorok (mint PLTL):
  - F p, G p, X p, p U q

- Állapotokon értelmezett operátorok:
  - A: „for All futures”,
    minden lehetséges útra az
    adott állapotból kiindulva



  - E: „Exists future”, „for some future”,
    legalább egy útra az
    adott állapotból kiindulva



Ismétlés: Elágazó idejű temporális logika: CTL

- Állapotokon értelmezhető összetett operátorok:
  - EX p: létezik útvonal, aminek következő állapotán p
  - EF p: létezik útvonal, aminek egy állapotán p
  - EG p: létezik útvonal, aminek minden állapotán p
  - E(p U q): létezik útvonal, amin p amíg q

  - AX p: minden útvonal következő állapotán p
  - AF p: minden útvonal egy-egy elérhető_állapotán p
  - AG p: minden útvonal minden állapotán p
  - A(p U q): minden útvonalon p amíg q

Az előadás áttekintése

Hogy működik a modellellenőrzés?

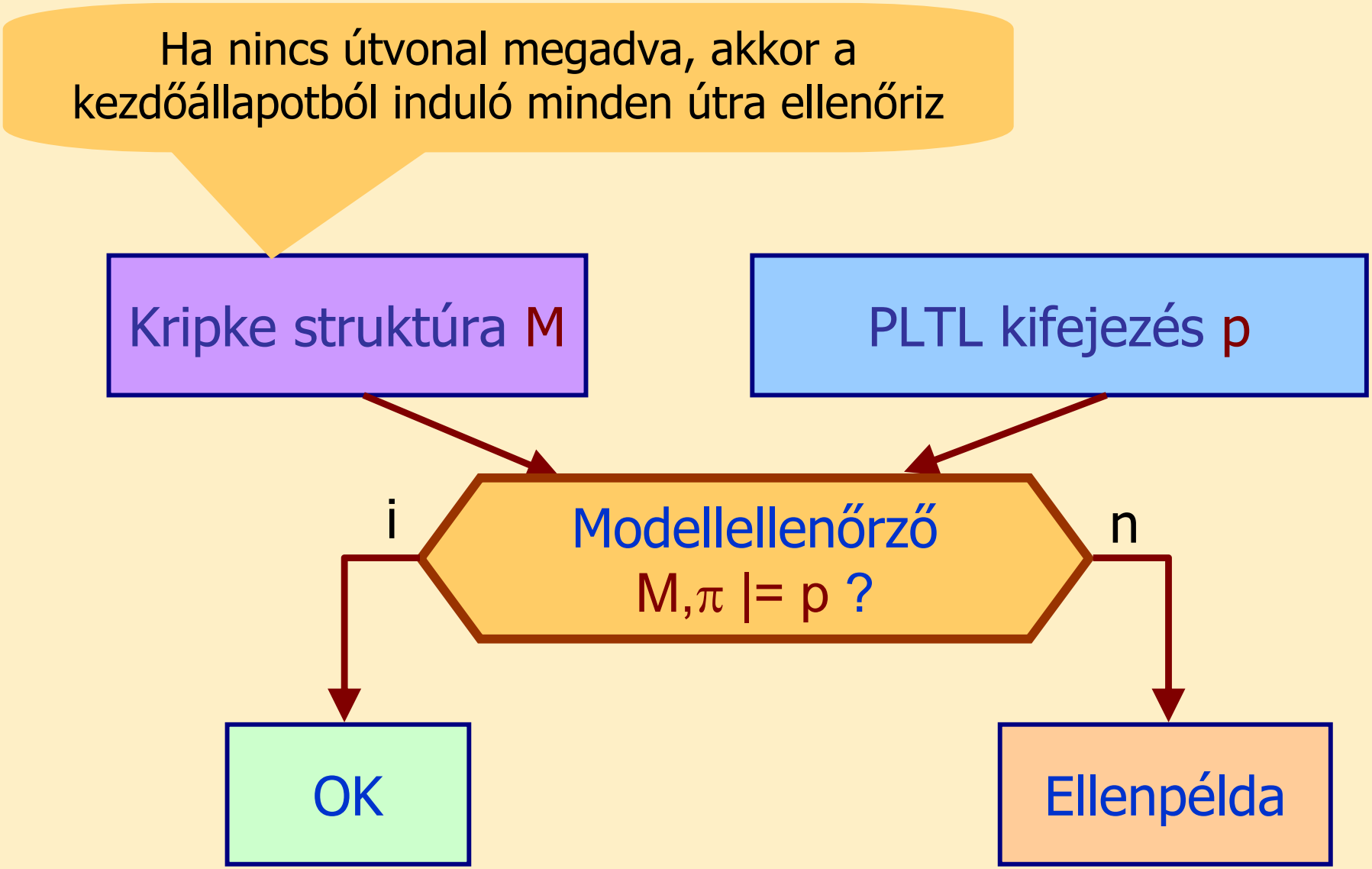
- A modellellenőrzés technikái
  - PLTL modellellenőrzés: Tabló módszer
  - CTL modellellenőrzés: Szemantika alapú módszer

Miért jó ezt tudni?

- Lehetőségek felmérése
  - Korlátok (ellenőrizhető modellek)
  - Optimalizálási lehetőségek (ld. következő előadás!)
- Érdekes alkalmazások
  - Automatikus teszt generálás
  - Alkalmazás a hardver tervezésben

PLTL modellellenőrzés: Tabló módszer

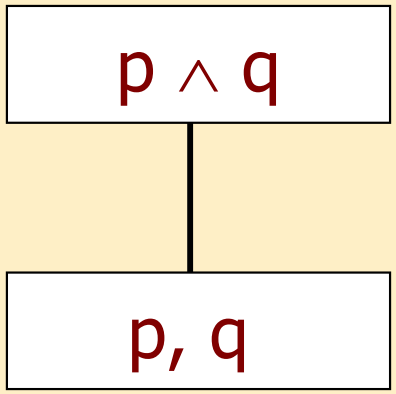
A modellellenőrzés feladata



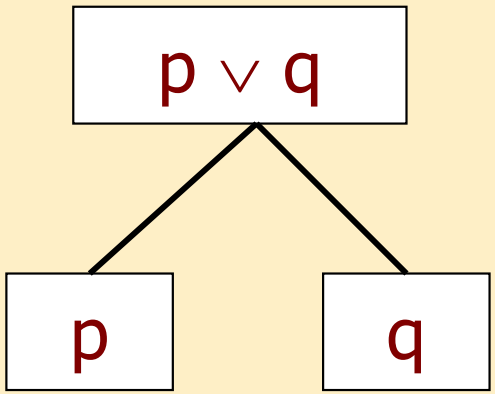
Bevezető: Tabló módszer a Boole logika esetén

Kérdés: Hogyan tehető igazzá egy adott kifejezés (Boole-függvény)?

- Alapötlet: A logikai kifejezés felbontása egy fa struktúrában (ez a tabló)
  - Csomópontok: Kifejezések, amelyeket igazzá akarunk tenni
  - Élek képzése: Konstrukciós szabályokkal az operátorok jelentése alapján; Elágazás megjelenik: Ha többféleképpen igazzá tehető egy kifejezés
- A konstrukció előtt a kifejezést negált normál formára kell hozni: Negálás ne legyen összetett kifejezések előtt
  - de Morgan azonosságok: $\neg(p \vee q)=(\neg p)\wedge(\neg q)$, $\neg(p \wedge q)=(\neg p)\vee(\neg q)$
- Tabló konstrukciós szabályok Boole logika esetén:



$$\frac{p \wedge q}{p, q}$$



$$\frac{p \vee q}{p \mid q}$$

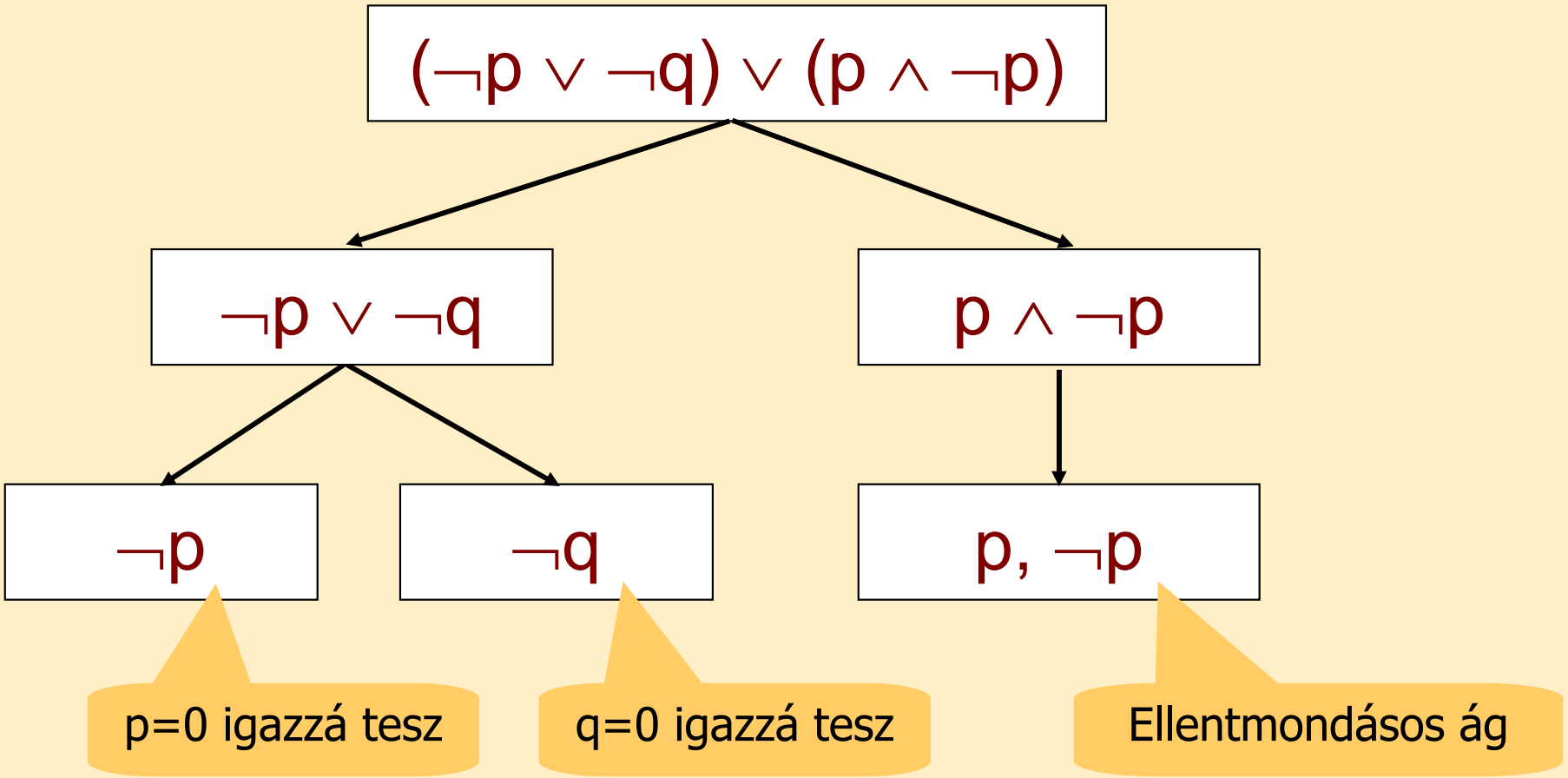
Bevezető: A tabló kiértékelése a Boole logika esetén

Meddig folytassuk a felbontást?

- Egy ág (felbontás) terminálása:
  - Ponált vagy negált változók listája marad a csomópontban, operátor nem
  - A lista minden elemét igazzá kell tenni a változók behelyettesítésével (a változók igaz vagy hamis értéket kaphatnak)
- Egy-egy ág terminálása után:
  - Ellentmondásos ág: Ugyanaz a változó ponált és negált formában is előfordul; nincs konzisztens behelyettesítés
    - Pl. p, $\neg p$ esetén ellentmondás, egyszerre nem lehet igaz a két kifejezés
  - Sikeres ág: Nincs ellentmondás; a lista minden eleme igazzá tehető behelyettesítéssel
    - Pl: p, $\neg q$ esetén: p igaz, q hamis a behelyettesítés, ami igazzá teszi
    - Az így adódó behelyettesítéssel a kezdeti kifejezés igazzá tehető
- A fa sikeres ágai jelölik ki, hol igaz a kifejezés

Bevezető: Egy tabló konstruálása Boole logika esetén

- Eredeti kifejezés: $\neg(p \wedge q) \vee \neg(\neg p \vee p)$
- Negálás bevitele: $(\neg p \vee \neg q) \vee (p \wedge \neg p)$
- Tabló konstruálás:



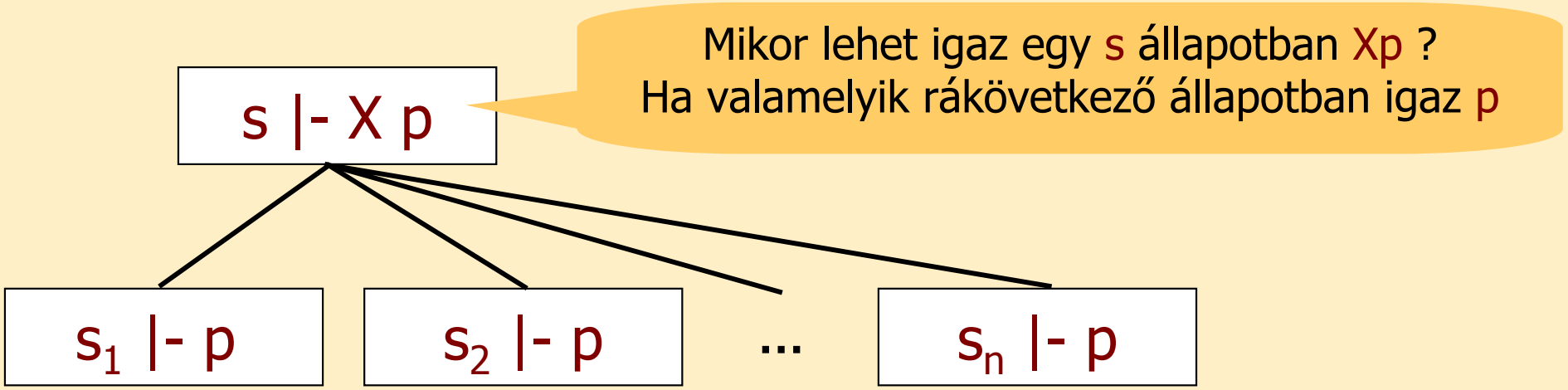
A tabló kiterjesztése PLTL-re

- Modellellenőrzés: Ellenpéldát keres, tehát

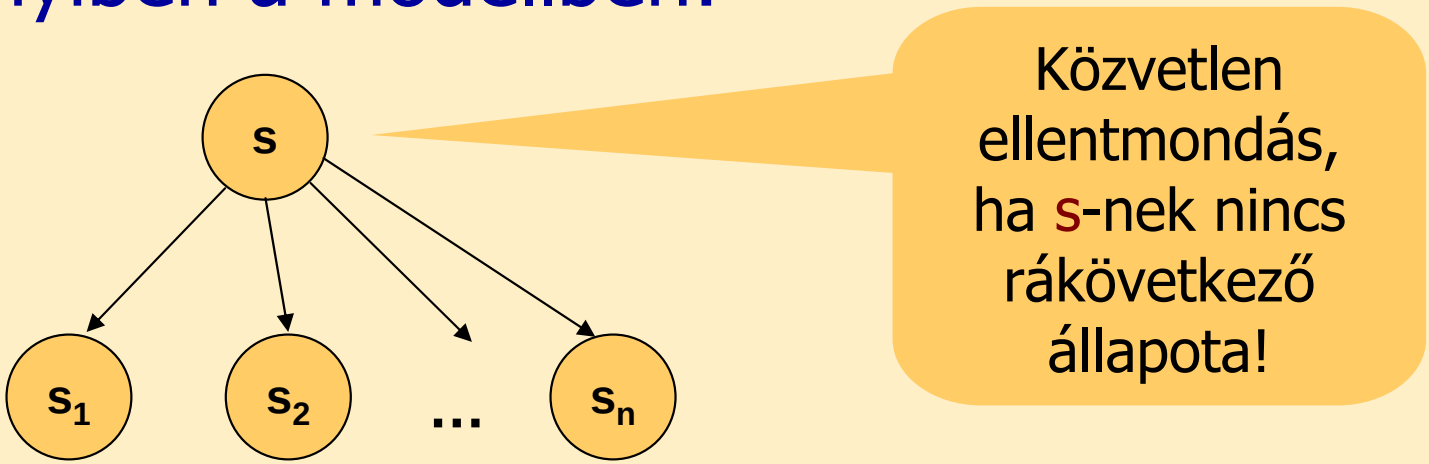
  negált kifejezésből készül a tabló!

  - Ha van sikeres (nem ellentmondásos) ág, az ellenpéldát ad!
  - Ha minden ág ellentmondásos, akkor az eredeti kifejezés igaz!
- Új felbontási szabályok kellenek a temporális operátorokhoz
  - Újdonság: A felbontás a modell alapján végezhető
  - Jelölés: s |- p jelöli, hogy p igazságát keressük s állapotból indulva
- Atomi kijelentések kezelése:
  - s |- P sikeres, ha $P \in L(s)$
  - s |- P ellentmondásos, ha $P \notin L(s)$
  - s |- $\neg$P sikeres, ha $P \notin L(s)$
- Temporális operátorok:
  - X és U felbontása elég (a többiek ezekkel kifejezhetők)
  - A szabályok a modellre fognak utalni

Felbontás az X operátor esetén

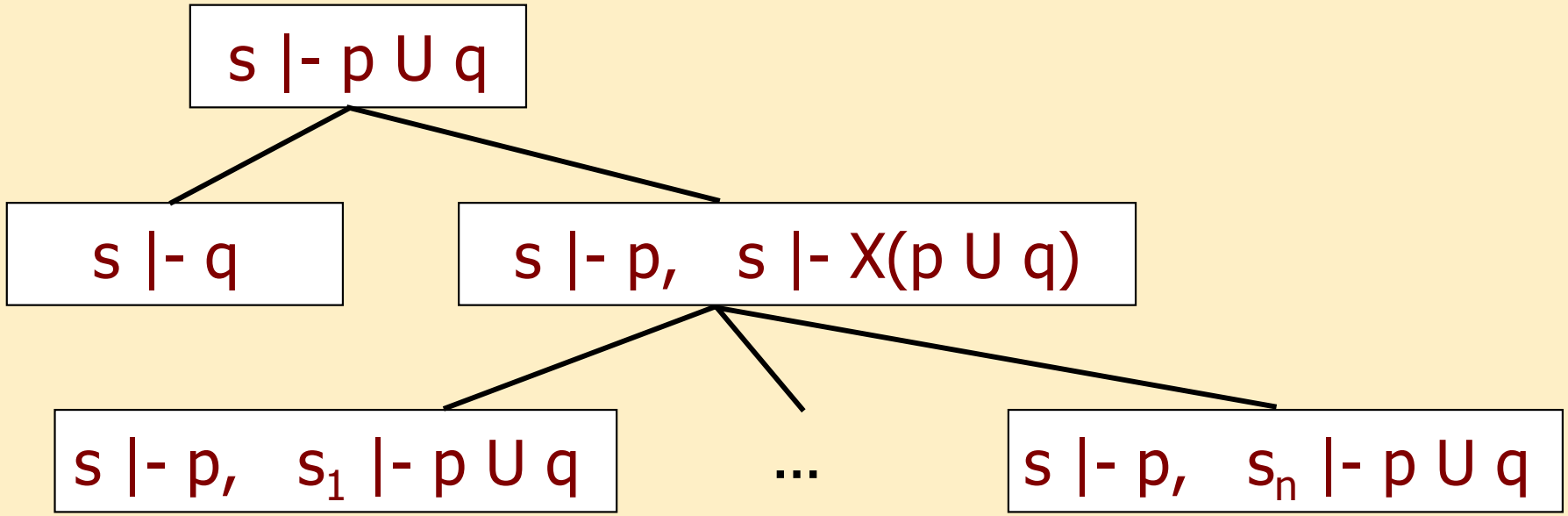


amennyiben a modellben:



Felbontás az U operátor esetén

- Felhasználjuk: $p\ U\ q = q \vee (p \wedge X\ (p\ U\ q))$

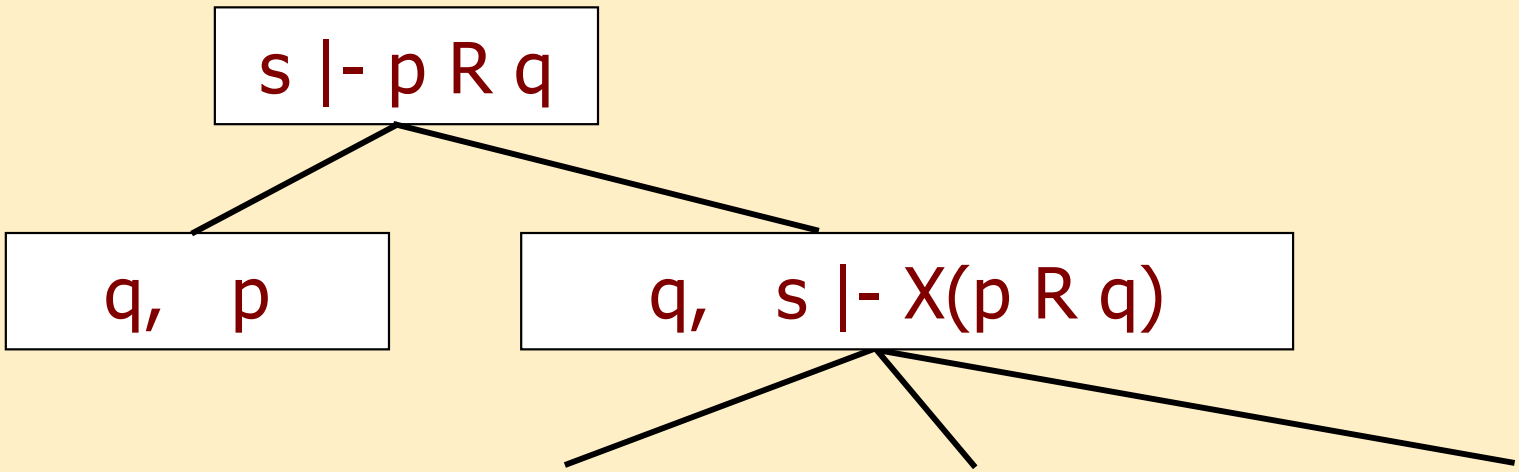


- A felbontás meddig folytatódik?
  - Ellentmondásra jutunk:
    - Útvonal végén X operátor van
    - Atomi kijelentésre vonatkozó lokális állítás nem teljesül
  - Sikeres ágak:
    - Atomi kijelentésekre vonatkozó állítás(ok) teljesül(nek)
    - Ciklust detektálunk, és nincs ellentmondás

Egy speciális operátor: R

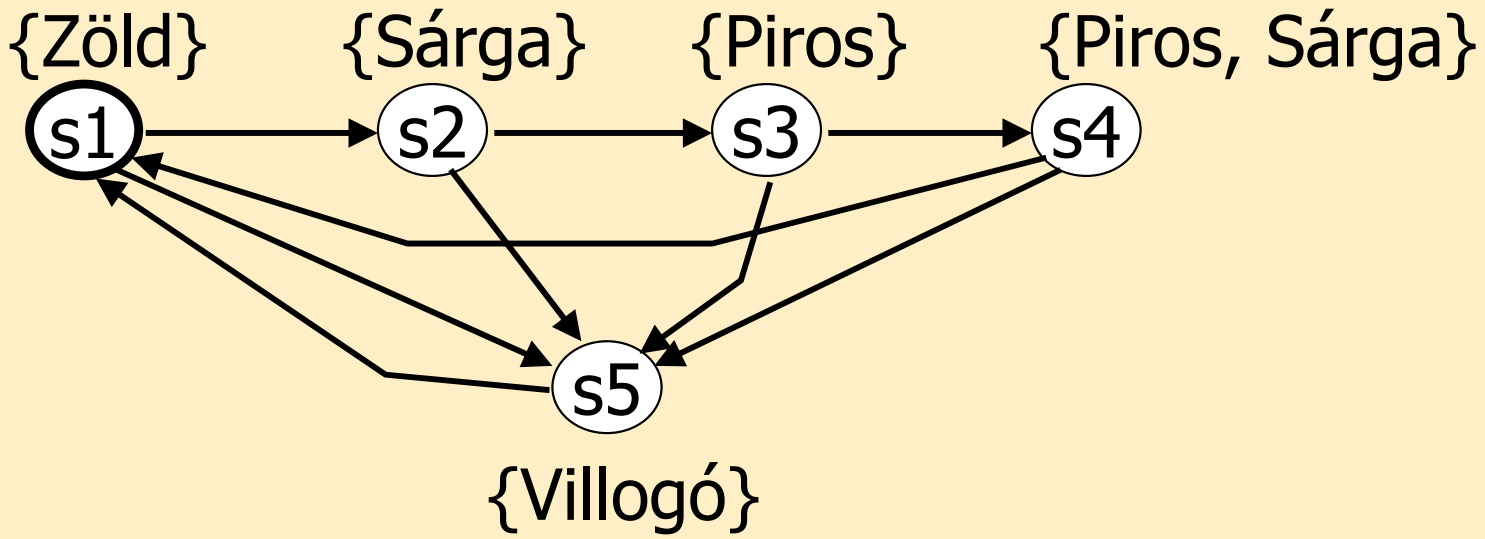
- Negált normál formára hozás az U operátor esetén:
  - $\neg(p\ U\ q) = ?$
  - Bevezethető az U operátor „duálisa”, az R (Release)
  - Definíció: $p\ R\ q = q \wedge (p \vee X\ (p\ R\ q))$
  - A negált normál formára hozás az R használatával:

    $\neg(p\ U\ q) = (\neg p)\ R\ (\neg q)$
- Az R operátor tablója:



Egy példa

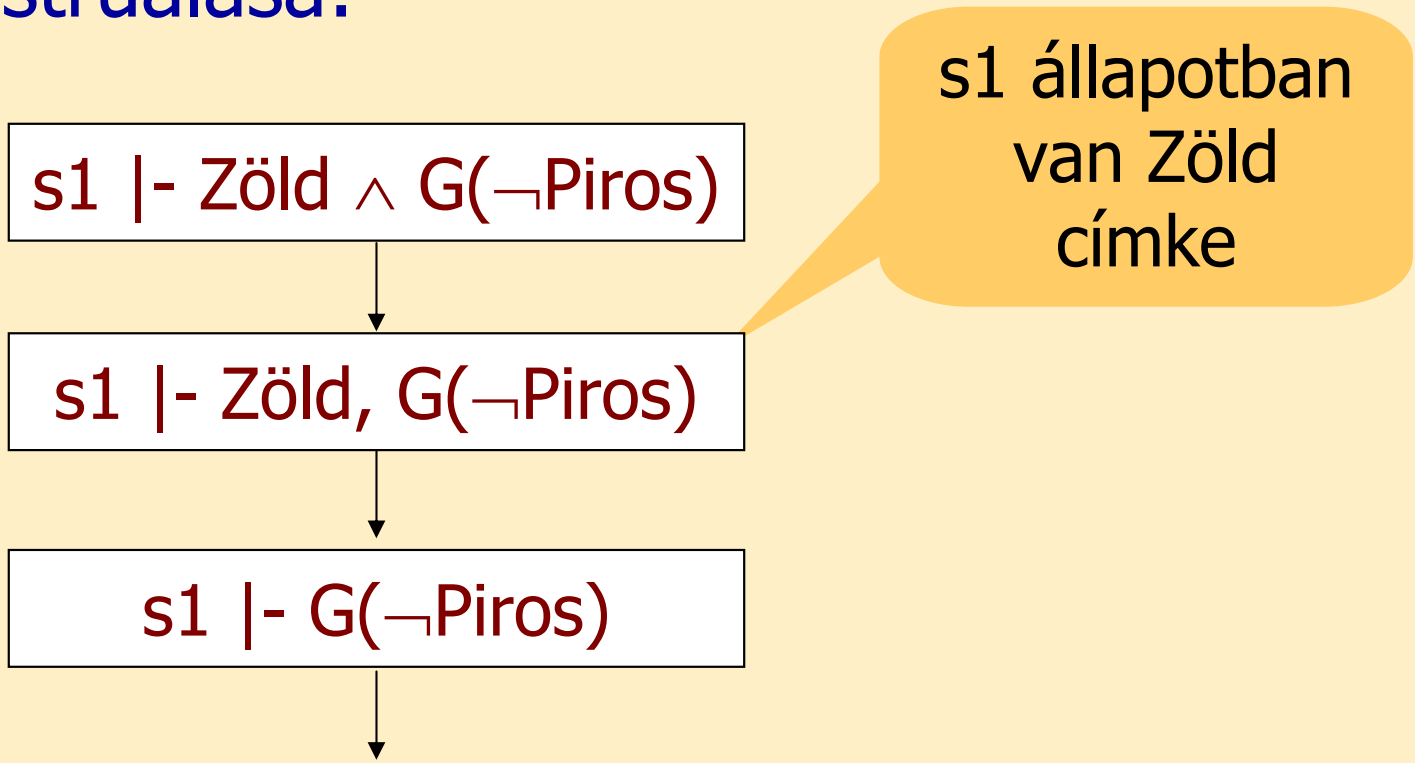
- A közlekedési lámpa módosított modellje (KS)
- Igaz-e, hogy ha a lámpa a kezdeti állapotban zöld, akkor előbb-utóbb piros lesz?
  - A PLTL kifejezés: Zöld $\Rightarrow$ F Piros



A kifejezés tablója

- A kifejezés negálása: s_1 |- ¬(Zöld ⇒ F Piros)
- Negált normál forma (ld. P ⇒ Q = ¬P ∨ Q):

  ¬(Zöld ⇒ F Piros) = Zöld ∧ ¬F Piros = Zöld ∧ G (¬Piros)
- A tabló konstruálása:



A kifejezés tablójának folytatása

s1 állapotban nincs Piros

s1 állapot után s2 vagy s5 jöhet

s2 állapotban nincs Piros

s2 után s3 vagy s5

Ellent-mondásos ág, s3-ban van Piros

s1 |- G(¬Piros)

s1 |- ¬Piros, XG(¬Piros)

s1 |- XG(¬Piros)

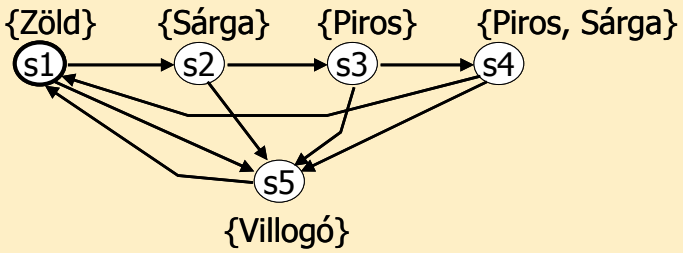
s2 |- G(¬Piros)

s2 |- ¬Piros, XG(¬Piros)

s2 |- XG(¬Piros)

s3 |- G(¬Piros)

s3 |- ¬Piros, XG(¬Piros)



s5 |- G(¬Piros)

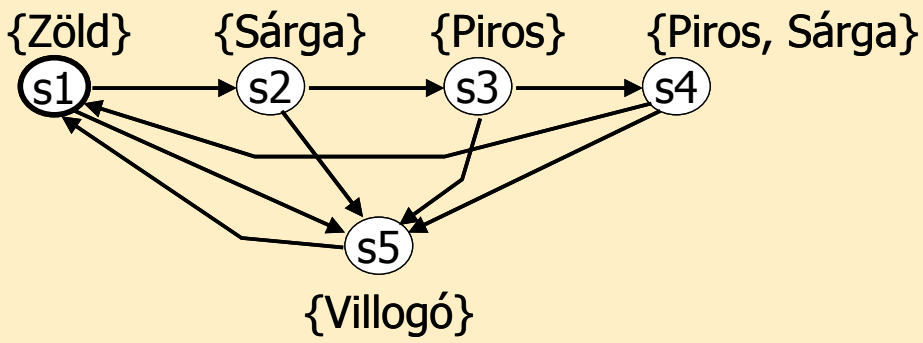
s5 |- ¬Piros, XG(¬Piros)

s5 |- XG(¬Piros)

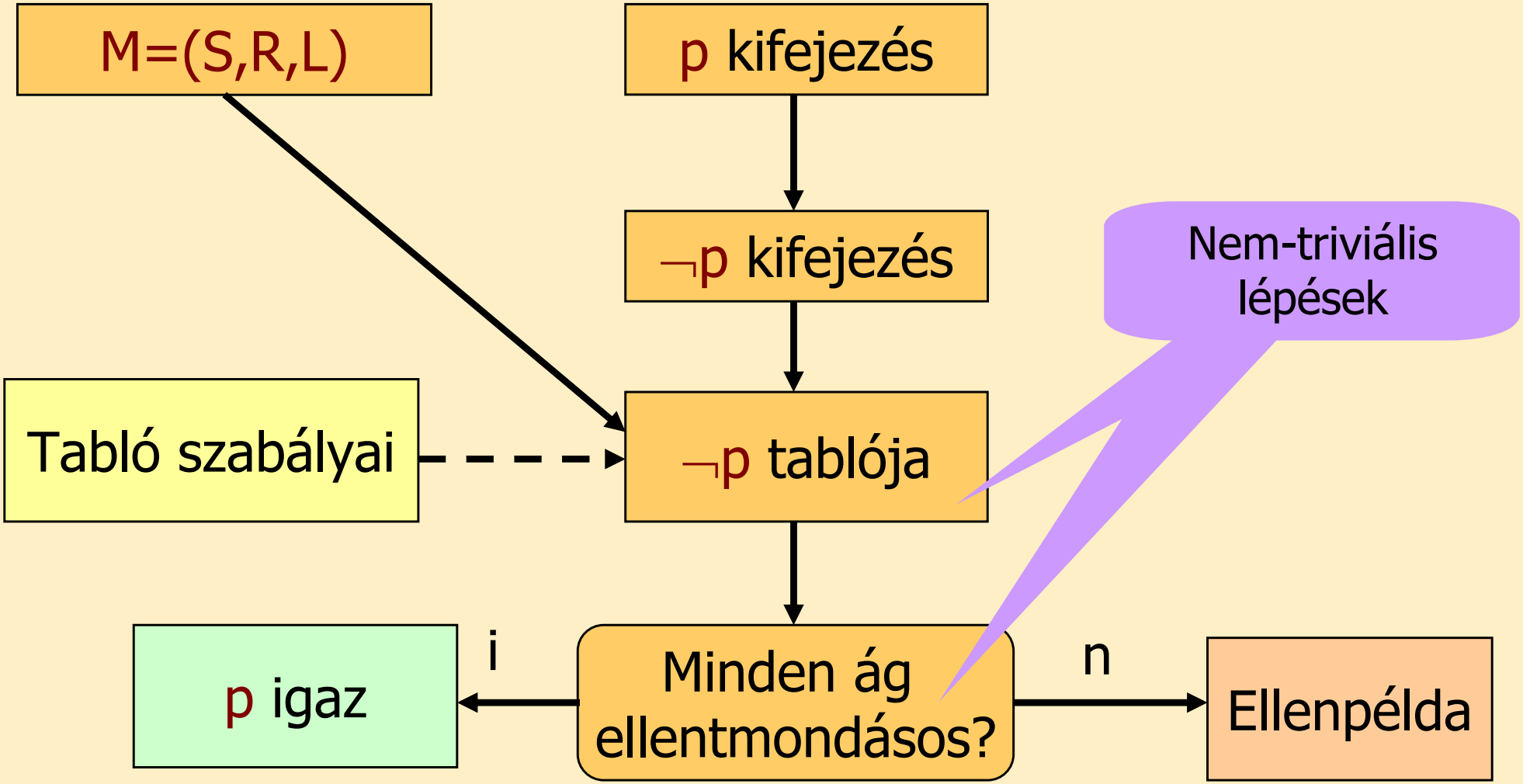
Ciklus (kétféleképpen), és nincs benne ellentmondás!

A modellellenőrzés eredménye

- A tabló eredménye:
  - Egy ellentmondásos ág
  - Két ellentmondás nélküli ciklus
- Következtetés:
  - Vannak olyan lefutások, ahol a negált kifejezés teljesül:
    Ciklus 1: s1, s2, s5, s1, …
    Ciklus 2: s1, s5, s1, …
- Az eredeti kifejezés Zöld $\Rightarrow$ F Piros tehát nem igaz
  - Ellenpéldák adhatók



A tabló módszer összefoglalása



PLTL modellellenőrzés: Automata alapú megközelítés (kiegészítő anyag)

Automaták véges hosszúságú szavakon

- $A=(\Sigma, S, S_0, \rho, F)$ ahol
  - Σ az ábécé, S állapotok, S_0 kezdőállapotok
  - ρ az állapotátmeneti reláció, $\rho: S \times \Sigma \rightarrow 2^S$
  - F az elfogadó állapotok halmaza.
- Az automata futása:
  - Egy beérkező $w=(a_0, a_1, a_2, \ldots a_n)$ betűsorozat hatására egy $r=(s_0, s_1, s_2, \ldots s_n)$ állapotsorozat
  - r elfogadó futás, ha $s_n \in F$
  - Egy w szót elfogad az automata, ha létezik rá elfogadó futás
- $L(A)=\{ w \in \Sigma^* \mid w \text{ elfogadott} \}$ az automata által elfogadott nyelv

Automaták végtelen hosszúságú szavakon

- Alkalmazás: Folyamatosan működő rendszerek
  - Nem ellenőrizhető, hogy a végállapot elfogadó-e
- Büchi elfogadási kritérium:
  - Egy beérkező $w=(a_0, a_1, a_2, \dots)$ betűsorozat hatására egy $r=(s_0, s_1, s_2, \dots)$ állapotsorozat
  - $\lim(r)=\{s \mid s$ előfordul végtelenül sokszor, azaz nincs olyan j, hogy $\forall k>j: s \neq s_k\}$
  - Elfogadó a futás, ha $\lim(r) \cap F \neq 0$
  - Egy w szót elfogad az automata, ha létezik rá elfogadó futás (azaz végtelen sokszor érint elfogadó állapotot)
- $L(A)=\{w \in \Sigma^* \mid w$ elfogadott$\}$ az automata által elfogadott nyelv

Az automata alapú módszer

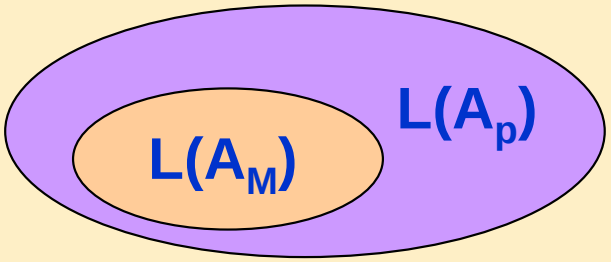
- A KS egy s állapotához: L(s) a betű a 2^{AP} ábécéből
  - Pl. {Piros, Sárga} az ábécé egy betűje
- A $\pi=(s_0, s_1, s_2, \ldots s_n)$ útvonal egy szót azonosít:

  $(L(s_0), L(s_1), L(s_2), \ldots L(s_n))$

- Két automatát kell konstruálni:
  - M=(S,R,L) alapján egy A_M automata konstruálható, amely azokat és csakis azokat a szavakat fogadja el, amelyek megfelelnek M útjainak.
  - p kifejezés alapján egy A_p automata konstruálható, amely azokat és csakis azokat a szavakat fogadja el, amelyek olyan utakhoz tartoznak, ahol p igaz
    - Itt felhasználhatók a tabló képzés szabályai:
      Nem a modell alapján bontunk, hanem állapotokat generálunk

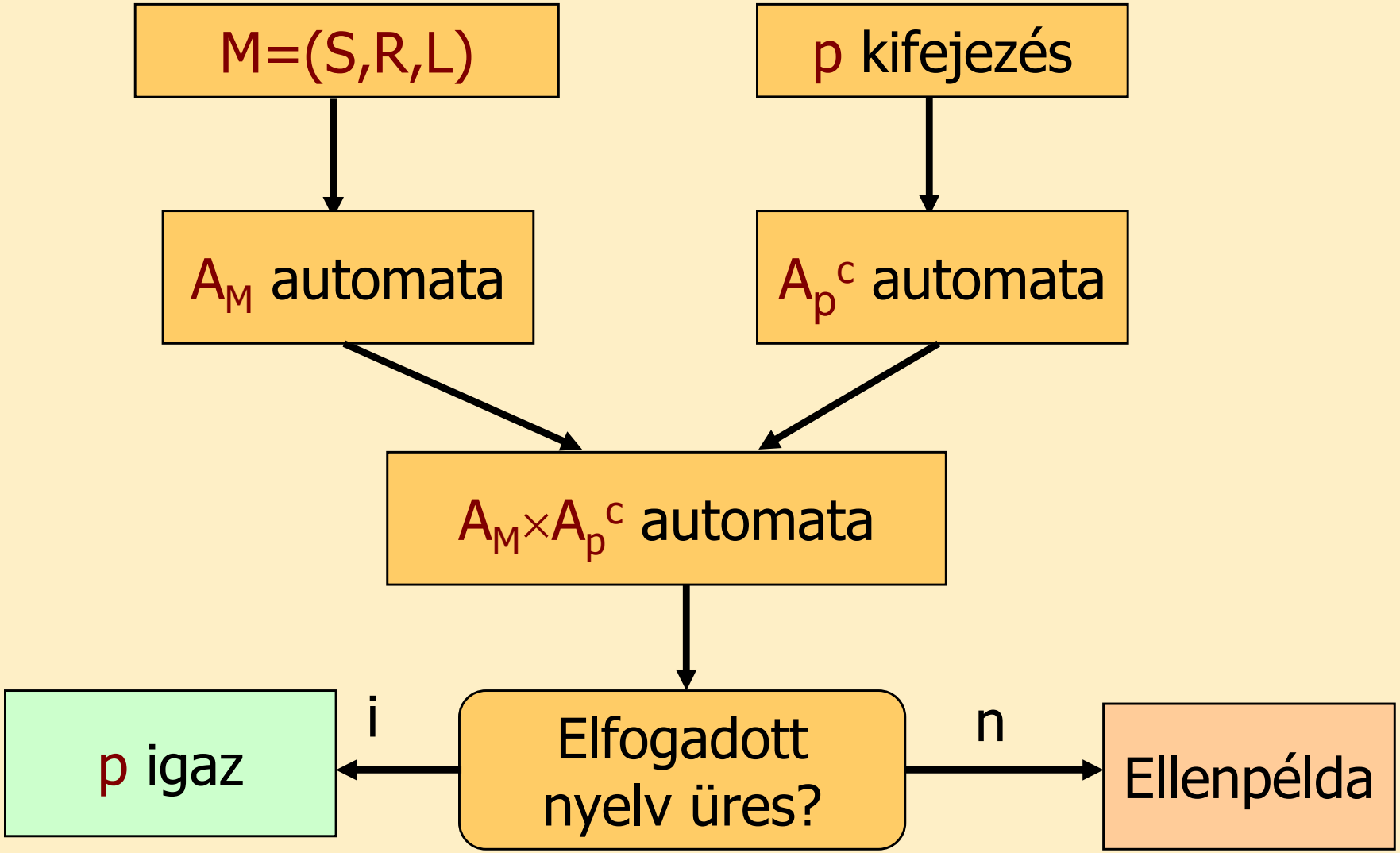
Modellellenőrzés az automatákkal

- Kérdés: $L(A_M) \subseteq L(A_p)$, vagyis a „modell” nyelv része-e a „tulajdonság” nyelvnek?
  - Ha igen, akkor $M \models p$



- A kérdés átalakítása:
  - Nyelvek metszetének ürességét kell vizsgálni:
    $L(A_M) \cap L(A_p)^c = 0$, itt $L(A_p)^c$ a komplementer nyelv
  - Az A_M „modell automata” és az A_p^c „komplementer tulajdonság automata” $A_M \times A_p^c$ szinkron szorzatát képezve, az általa elfogadott nyelv üres-e?
    - Ha üres, akkor $M,\pi \models p$ teljesül
    - Az elfogadott nyelv üres, ha nincs elérhető elfogadó állapot
- Folyamatosan működő rendszerek:
  - Automaták végtelen hosszúságú szavakon;
    Büchi elfogadási kritérium: cikluskeresésre vezet

Az automata alapú modellellenőrzés

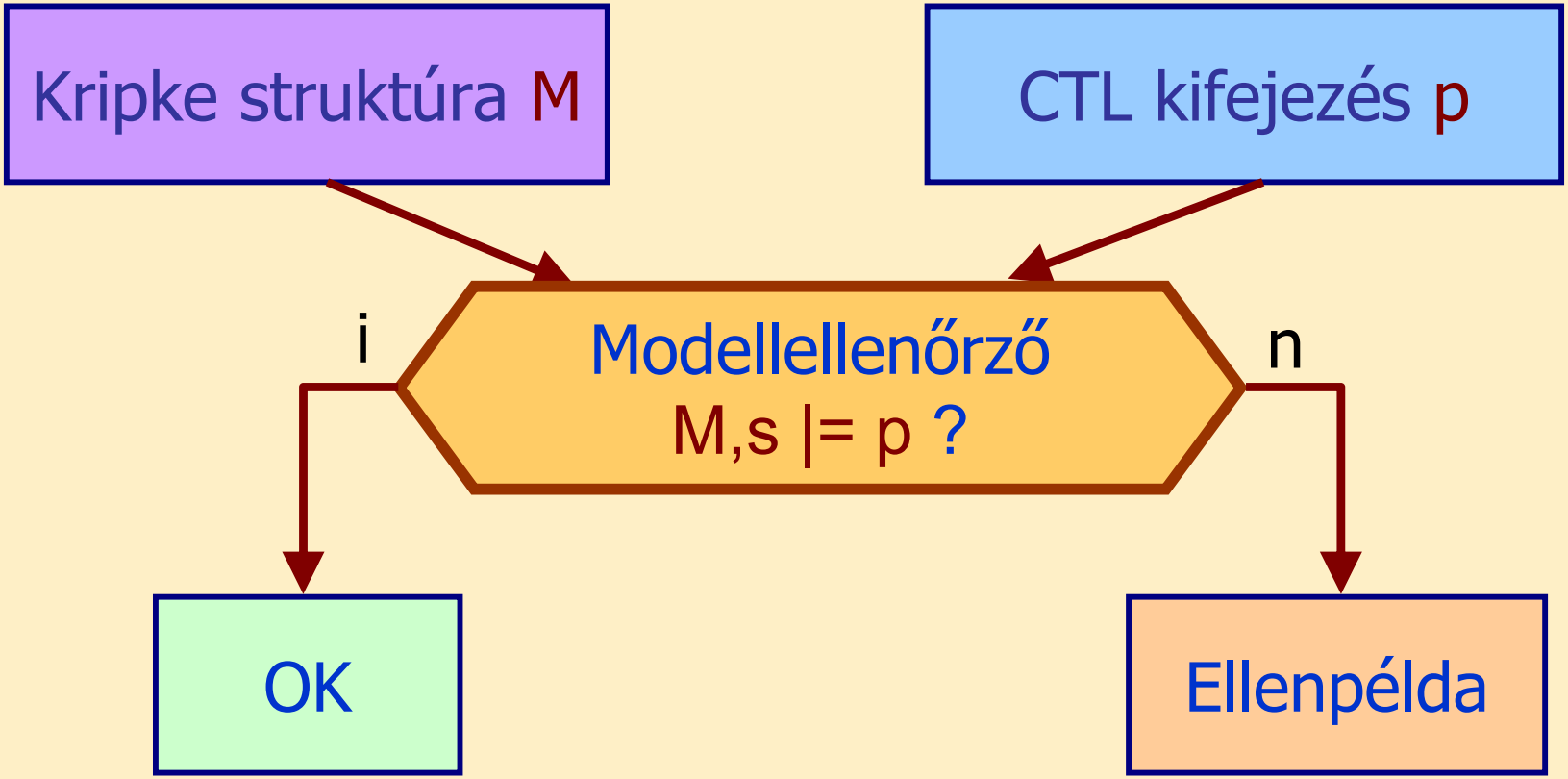


„On-the-fly” modellellenőrzés

- Alapötlet:
  - Az A_p automata generálása közben lehet elvégezni a szinkron szorzat automata konstruálását is
- Szinkron szorzat automata konstruálása:
  - Az ellenőrizendő kifejezés által vezérelten történik: ahogyan az A_p automata új állapota előáll, úgy kell A_M állapotait „előkeresni”
  - Nem szükséges hozzá a modell állapottér teljes generálása
    - Pl. egy magasabb szintű modellből való származtatás esetén

CTL modellellenőrzés: Szemantika alapon

A modellellenőrzés feladata



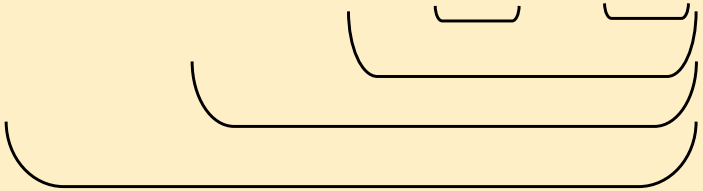
Alapötlet: Állapotok címkézése

- Globális modellellenőrzés:
  - Jelölés: Sat(p) jelöli egy p CTL kifejezés esetén azoknak az állapotoknak a halmazát, ahol p igaz
  - Címkézés: Ezeket az állapotokat p-vel címkézzük
  - Ezután $s \in Sat(p)$ egyszerűen vizsgálható egy adott s állapotra: Szerepel-e rajta a p címke?
- A címkézés (azaz Sat(p) számítása) inkrementálisan történik
  - Címkézett állapothalmazok bővítése
  - Az iteráció vége: Nem nő a címkézhető állapotok halmaza

CTL modellellenőrzés állapot címkézéssel

- Állapotok címkézése: Hol igaz egy adott kifejezés
- Összetett kifejezés esetén hogyan történik a címkézés?
  - Kifejezések felbontása azok struktúrája alapján, és „belülről kifelé” Sat(kifejezés) számítások:

$$AF\ (\ P \wedge E\ (Q\ U\ R))$$



- Algoritmus az összetett kifejezés felbontása alapján:
  - Kiindulás: KS címkézve van atomi kijelentésekkel
  - Továbblépés: Címkézés az összetettebb kifejezésekkel
    - Ha p illetve q címkék már vannak, akkor megadható, hol lehet $\neg p$, $p \wedge q$, EX p, AX p, E(p U q), A(p U q) címke
    - Ehhez felhasználhatók a szemantika szabályai
    - Inkrementális címkézés történik

Hol igaz egy adott kifejezés?

- P (atomi kijelentés) azokban az s állapotokban igaz, ahol $P \in L(s)$
  - Itt P címkeként már szerepel a KS-ba
- $\neg P$ azokban az s állapotokban igaz, ahol $P \notin L(s)$
  - Ezek az állapotok $\neg P$ kifejezéssel címkézhetők
- $p \wedge q$ azokban az s állapotokban igaz, ahol p és q is igaz
  - Egy állapot címkézése lehet $p \wedge q$, ha címkéi között már van p és q
- Temporális operátorok: EX, AX, E(U), A(U)
  - Bonyolultabb algoritmust igényel a címkézés!

Az AX, EX alakú kifejezések

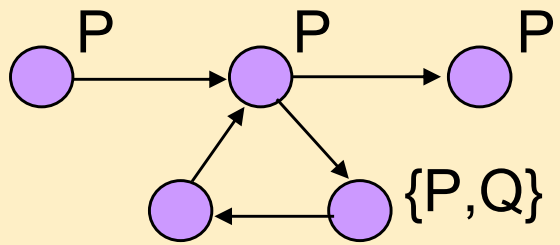
- EX p azokban az s állapotokban igaz, amelyeknek van olyan rákövetkező állapota, ahol p igaz
  - Egy állapot címkézése lehet EX p, ha van olyan rákövetkező állapota, ami p-vel címkézett

- AX p azokban az s állapotokban igaz, amelyeknek minden rákövetkező állapotában p igaz
  - Egy állapot címkézése lehet AX p, ha minden rákövetkező állapota p-vel címkézett

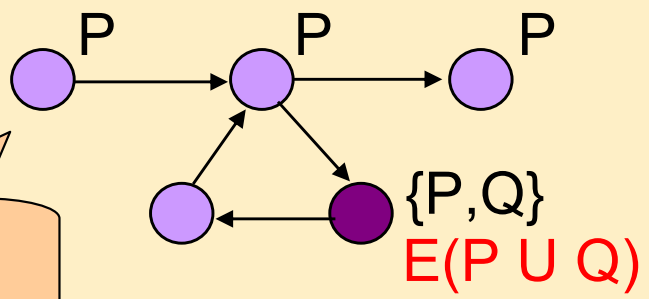
Az E(p U q) kifejezések

- Hol igaz E(p U q)?
  - Azonosság: $E(p\ U\ q) = q \vee (p \wedge EX\ E(p\ U\ q))$
  - „Rekurzív” képlet…
- Tehát mely s állapotok címkézhetők E(p U q)-val?
  - Ha s címkézett q-val, vagy
  - ha s címkézett p-vel, és legalább egy rákövetkezője már címkézett E(p U q)-val
- Iteráció adódik:
  - q-val már címkézett állapotok adják azokat az állapotokat, ahol először megjelenik az E(p U q) címke
  - Ezek megelőző állapotait kell végignézni: Ha szerepel ott a p címke, akkor rátehető az E(p U q) címke is!
  - Így visszafelé járjuk be azokat az útvonalakat, amik p-vel címkézett állapotokon keresztül visznek q-val címkézett állapotba

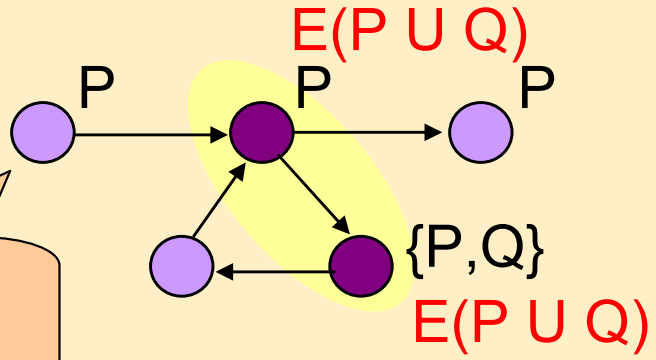
Az E(P U Q) címkézés iterációja



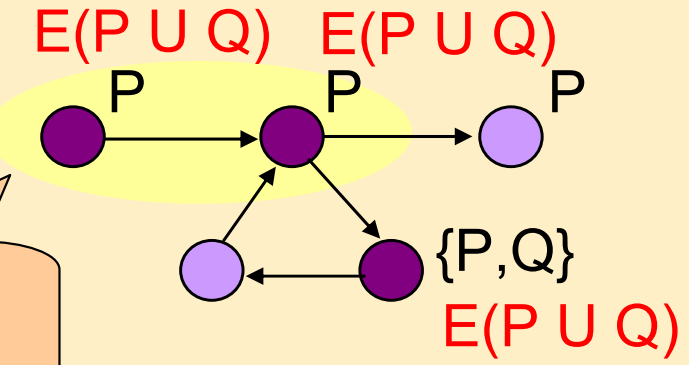
Kripke struktúra a kezdő címkézéssel



Első lépés: Q



Második lépés: P ∧ EX



Harmadik lépés: P ∧ EX

- Az iteráció addig tart, míg nő az állapothalmaz (fixpontot érünk el)

Az A(p U q) kifejezések

- Hol igaz A(p U q)?
  - Azonosság: $A(p\ U\ q) = q \vee (p \wedge AX\ A(p\ U\ q))$
  - Ez is „rekurzív” képlet…
- Tehát mely s állapotok címkézhetők A(p U q)-val?
  - Ha s címkézett q-val, vagy
  - ha s címkézett p-vel, és minden rákövetkezője már címkézett A(p U q)-val
- Iteráció adódik:
  - q-val már címkézett állapotok adják azokat az állapotokat, ahol először megjelenik az A(p U q) címke
  - Ezek megelőző állapotait kell végignézni: Ha szerepel ott a p címke, és minden rákövetkező állapotukon szerepel az A(p U q) címke, akkor ezekre is rátehető az A(p U q) címke

Ezzel a formális szintaxisban használt operátorokat lefedtük!

Még egy példa az iterációra

- Hol igaz AF p?
  - Azonosság: AF p = p $\vee$ AX AF p
  - „Rekurzív” képlet…
- Tehát mely s állapotok címkézhetők AF p-vel?
  - Ha s címkézett p-vel, vagy
  - ha minden rákövetkezője már címkézett AF p-vel
- Iteráció adódik:
  - p-vel már címkézett állapotok adják azokat a kiinduló állapotokat, ahol először megjelenik az AF p címke
  - Ezek megelőző állapotait kell végignézni:
    Ha minden rákövetkező állapotukon szerepel az AF p címke, akkor rátehető az AF p címke
  - Így visszafelé keressük azokat az állapotokat, amik minden útvonalon p-vel címkézett állapotokba vezetnek

Iteráció halmazműveletekkel

- A címkézés bővítése halmazműveletekkel történik
  - Kezdőhalmaz: Rész-kifejezésekkel már címkézett állapotok
  - Címkézés bővítése:
    - E(p U q) esetén: „Legalább egy rákövetkező állapota már címkézett …”
    - A(p U q) esetén: „Minden rákövetkező állapota már címkézett …”
  - Ez alapján a megelőző állapotok valamelyikére tehető az új címke
- Hogyan definiálhatók ezek a halmazok formálisan?
  - Már címkézett Z állapothalmaz alapján:

    $pre_E(Z)=\{s\in S \mid$ létezik olyan s', hogy $(s,s')\in R$ és $s'\in Z\}$

    $pre_A(Z)=\{s\in S \mid$ minden s'-re, ahol $(s,s')\in R$: $s'\in Z\}$
- Példa: E(P U Q) iterációja:
  - Kezdőhalmaz: $Z_0 = \{s \mid Q\in L(s)\}$
  - Iteráció: $Z_{i+1}= Z_i \cup (pre_E(Z_i) \cap \{s \mid P\in L(s)\})$

Eddig címkézettek és

A megfelelő megelőző állapotok amelyek ...

... P-vel címkézettek

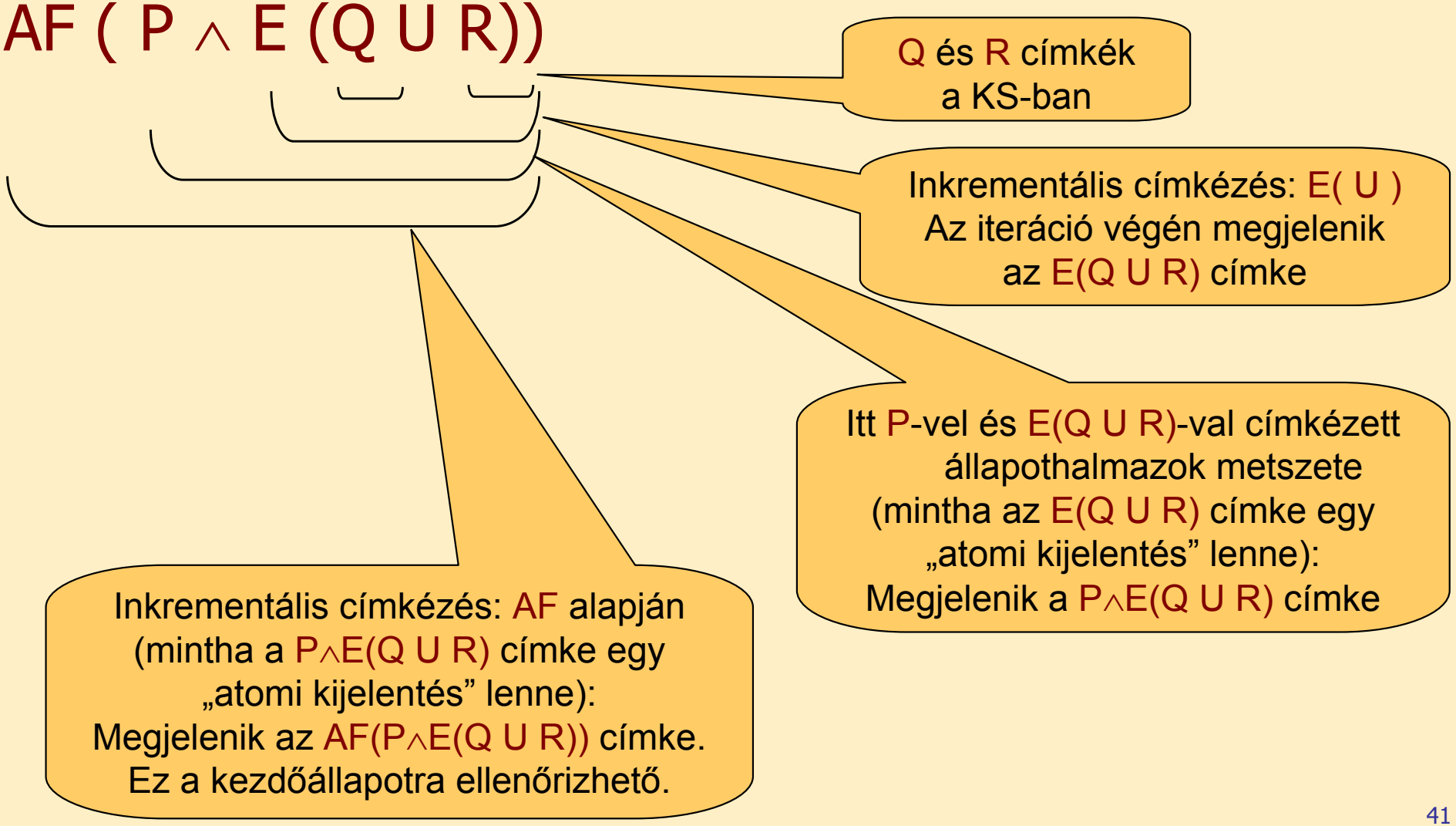
  - Iteráció vége: Ha $Z_{i+1}= Z_i$, azaz nem bővül a halmaz

CTL modellellenőrzés: Összefoglalás

- Globális modellellenőrzés:
  - Állapotok címkézése azokkal a (rész)kifejezésekkel, amelyek igazak az adott állapotban
- Kifejezések felbontása (szintaxis szabályok alapján)
  - Atomi kijelentésekből indítva az összetettebb kifejezések felé („belülről kifelé”)
  - Az előző lépésben adott címkézés felhasználása
- EX, AX esetén: Megelőző állapot vizsgálata
- E(p U q), A(p U q) esetén: Inkrementális címkézés
  - Kezdőhalmaz:
    - A belső kifejezések (p, q) által meghatározott állapothalmazok alapján
  - Iteráció: A szemantika alapján (megelőző állapotokra lépegetve)
  - Iteráció vége: Nem nő a címkézett állapotok halmaza
    - Precíz matematikai algoritmus: Fixpont számítás teljes hálókban

A bevezető példa kifejtése

- Kifejezések felbontása azok struktúrája alapján, és „belülről kifelé” címkézés:
  AF (P ∧ E (Q U R))



Összefoglalás

- PLTL modellellenőrzés
  - Tabló konstruálása
    - Boole-logikai bevezetés: Ellentmondásos és sikeres ágak
    - Tabló PLTL esetén: Ellenpélda keresés (negált követelmény)
  - Kiegészítés: Automaták szorzata
- CTL modellellenőrzés
  - Szemantika alapú modellellenőrzés
    - Inkrementális címkézés bővülő részkifejezésekkel (globális modellellenőrzés)
    - Halmazműveletekkel történik

Hogyan tehető hatékonnyá ez az algoritmus?