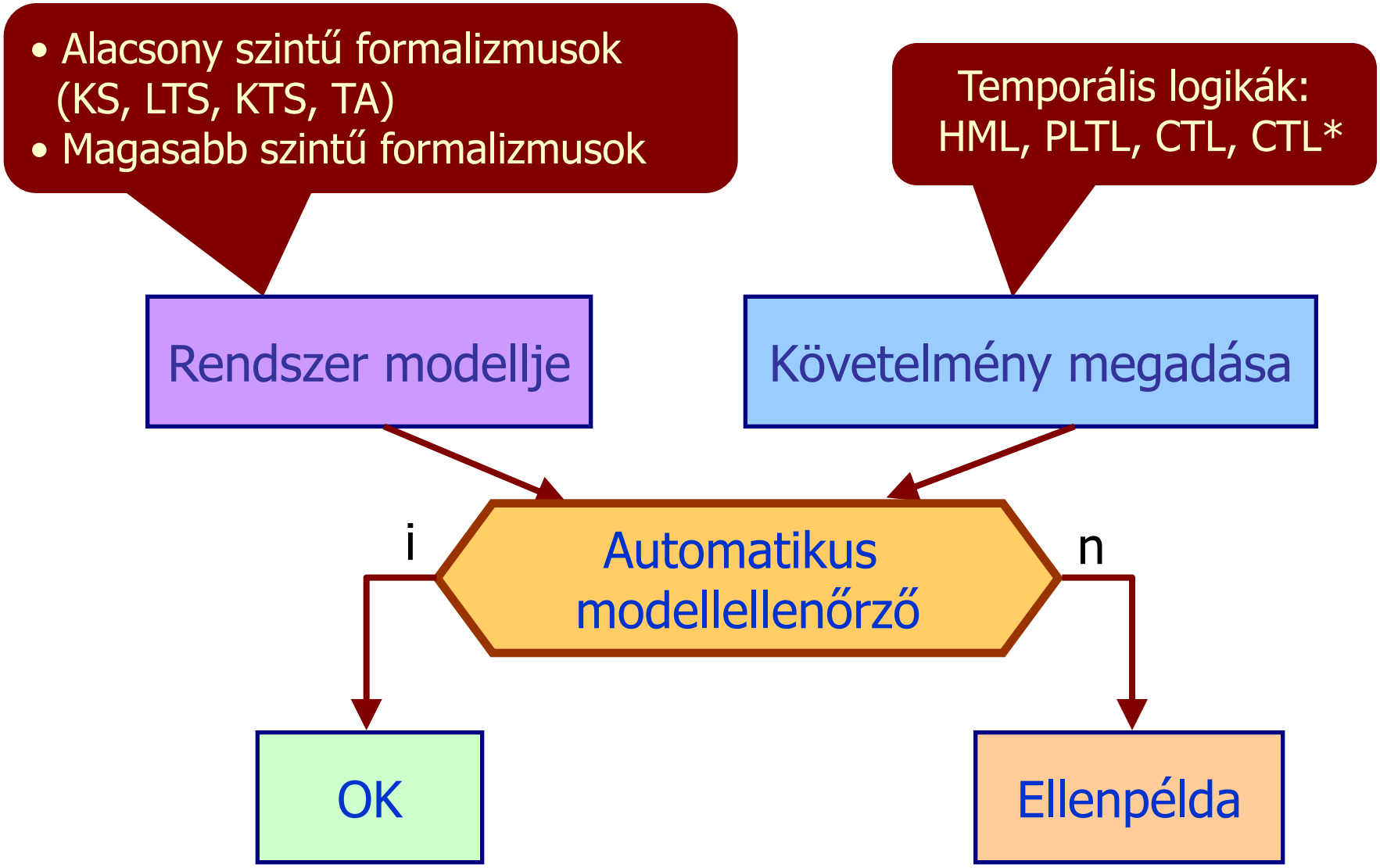
Hennessy-Milner logika. Lineáris temporális logika és modellellenőrzése

Majzik István
Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék
http://www.mit.bme.hu/~majzik/

Tartalomjegyzék

Motiváció: Mit szeretnénk elérni?



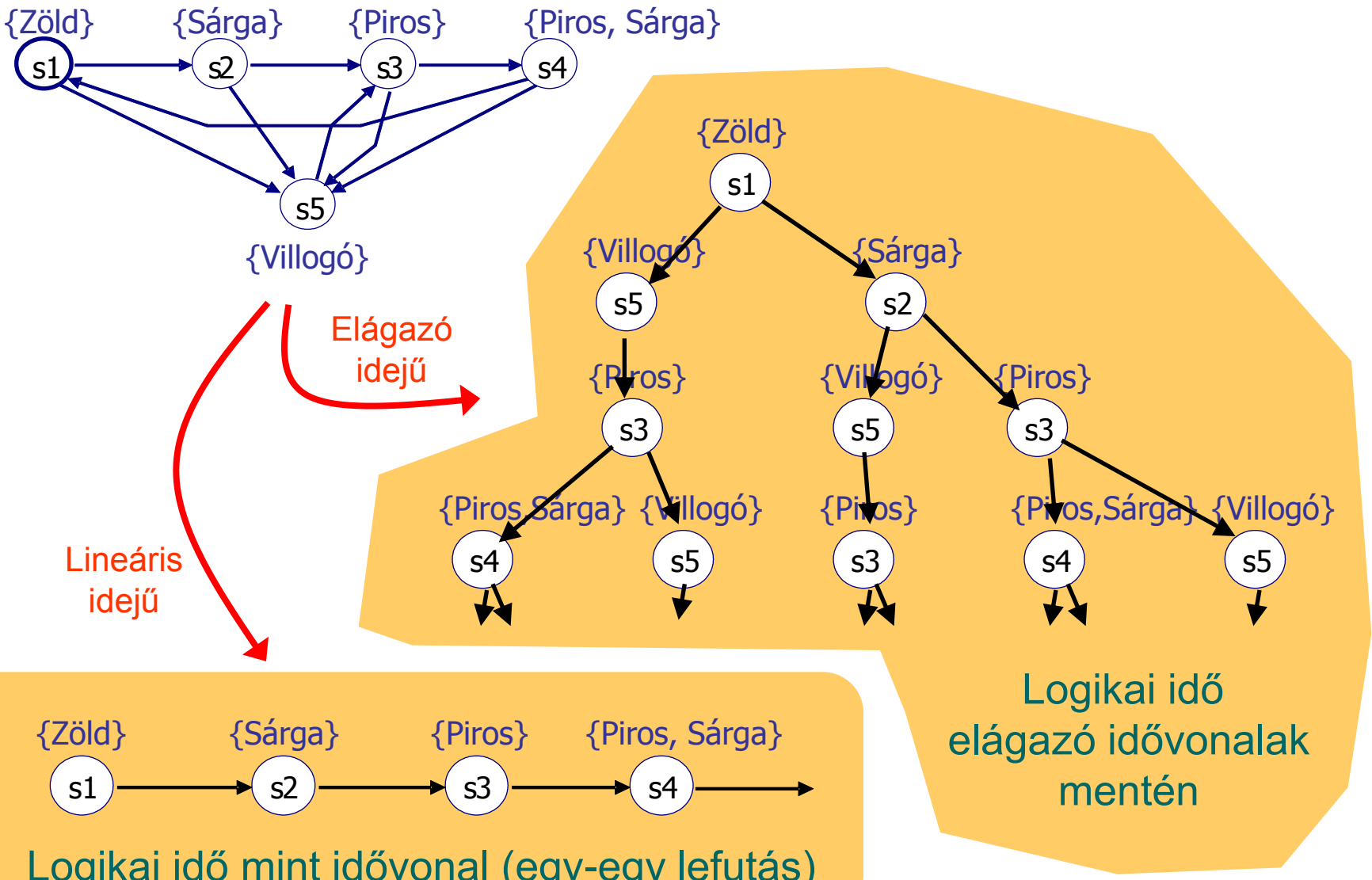
Tipikus követelmények

- Biztonsági követelmények:
  - Veszélyes, nemkívánatos helyzetek elkerülését fogalmazzák meg
    - Pl. holtpont, jogosulatlan hozzáférés, veszélyes álapot elkerülése
  - Invariáns jellegű tulajdonság,
    univerzális kvantor az elérhető állapotokon („mindig …”)
  - Ha egy állapotsorozat nem teljesíti, akkor
    nem is egészíthető ki úgy, hogy teljesítse
- Élőségi követelmények:
  - Kívánatos helyzetek elérését írják elő
    - Pl. eredmény, válasz, kiszolgálás bekövetkezik
  - Egzisztenciális kvantor az állapotokon („létezik …”)
  - Ha egy állapotsorozat nem teljesíti,
    akkor elvileg kiegészíthető úgy, hogy teljesítse

Követelmények leíró nyelve

- Elérhetőség: Több állapot bekövetkezési sorrendjére vonatkozó követelmények
  - Megfeleltethető a logikai időnek:
    - Jelen időpillanat: Aktuális állapot
    - Következő időpillanat(ok): Rákövetkező állapot(ok)
  - Temporális operátorok (logikai időbeli, sorrendi operátorok) használhatók a követelmények kifejezésére
- Temporális logikák:
  - Formális rendszer arra, hogy kijelentések igazságának logikai időbeli változását vizsgálhassuk
  - Temporális operátorok: „mindig”, „valamikor”, „mielőtt”, „addig, amíg”, „azelőtt, hogy”, …

Lineáris és elágazó idejű temporális logikák



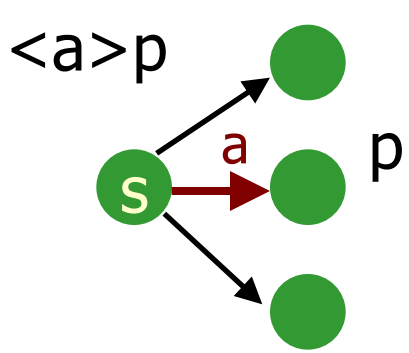
Modellellenőrzés

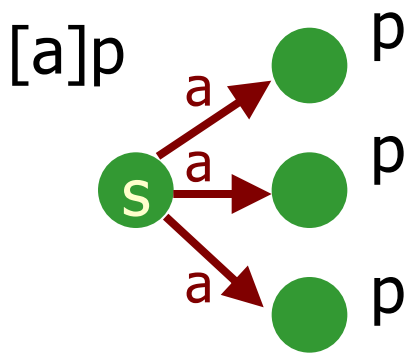
- Annak vizsgálata, hogy egy p temporális logikai kifejezés igaz-e az M modellen
- Fajtái:
  - Lokális: M adott kezdőállapotára
  - Globális: Minden (kezdő)állapot keresése, ami esetén p igaz lesz
- Technikák:
  - Szemantikán alapuló: Az operátorok szemantikája alapján „keresés” az állapotérben
  - Automata-elméleti: Visszavezetés az automata-elmélet eredményeire
  - Tabló módszer: Bizonyításkereséssel (a kifejezés felbontásával) kideríteni, hol lehet igaz a kifejezés

Tartalomjegyzék

A Hennessy-Milner logika

- LTS-en értelmezett egyszerű logika
- Véges akciószekvenciák (mint tulajdonságok) megadására alkalmas (pl. teszt lefutása)
- Szintaxis:
  HML ::= true | false | p∧q | p∨q | [a]p | <a>p
- A temporális operátorok intuitív jelentése:





Hennessy-Milner logika szemantikája

T=(S, Act, $\rightarrow$) LTS-en értelmezett

Jelölés: T,s |= p jelentése: T LTS s állapota esetén igaz p

- **H1**: T,s |= true, T,s $\not\models$ false
- **H2**: T,s |= p$\wedge$q a.cs.a. T,s |= p és T,s |= q
  T,s |= p$\vee$q a.cs.a. T,s |= p vagy T,s |= q
- **H3**: T,s |= [a]p a.cs.a. $\forall$s' ahol s $\rightarrow^a$ s': s' |= p
- **H4**: T,s |= <a>p a.cs.a. $\exists$s': s $\rightarrow^a$ s' és s' |= p

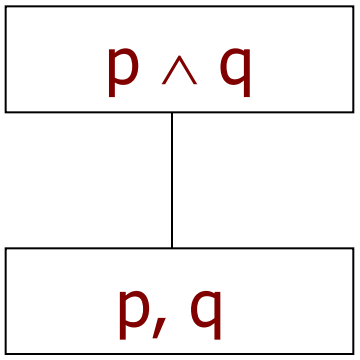
HML példák:

  - <a>true: igaz, ha létezik a-val címkézett kimenő átmenet
  - [a]false: igaz, ha nincs a-val címkézett kimenő átmenet
  - <a><b><c>true: igaz, ha létezik az a,b,c szekvencia

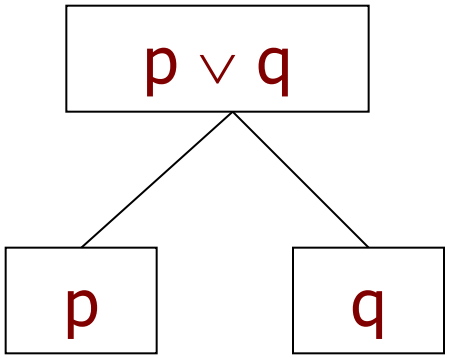
HML alapú modellellenőrzés: Tabló módszerrel

A tabló módszer bevezetése (Boole logika)

- A matematikai logika egy módszere
  - Arra a kérdésre válaszol, igazzá tehető-e egy adott kifejezés
- Tabló: Egy logikai kifejezés felbontása fa struktúrában
  - Csomópontok: (Rész-)kifejezések, amiket igazzá kell tenni
  - Élek: Rész-kifejezések viszonya (tabló konstrukciós szabály)
- A konstrukció előtt a kifejezést negált normál formára kell hozni (ld. de Morgan): Negálás csak változók előtt lehet
- Tabló konstrukciós szabályok Boole logika esetén:



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



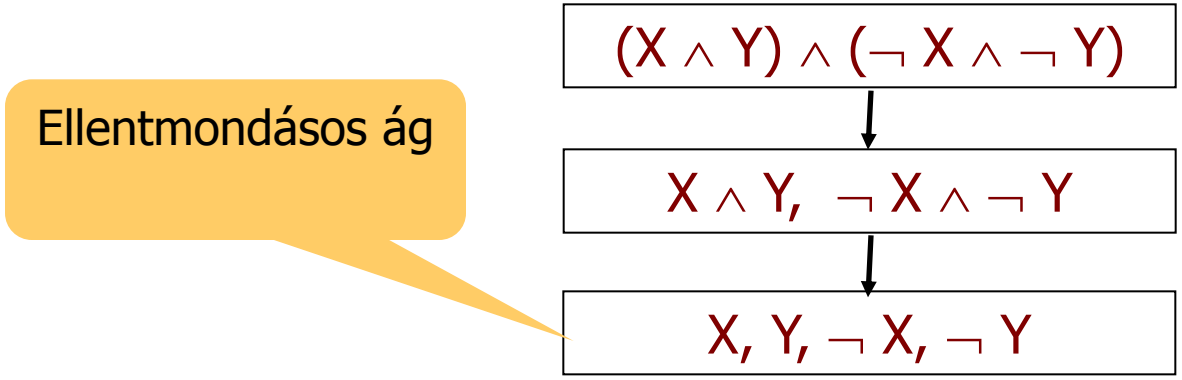
$$\frac{p \vee q}{p \mid q}$$

A tabló kiértékelése

- Egy ág (felbontás) terminálása: Csak (ponált vagy negált) változók maradhatnak a csomópontban
  - Ezek igaz vagy hamis értéket kaphatnak
- Egy ág terminálása után:
  - „Ellentmondásos” ág: Ugyanazon változó ponált és negált formában is előfordul, így nem lehet érvényes behelyettesítés
    - Pl. p, ¬p, q esetén ellentmondás
  - „Sikeres” ág: Nincs ellentmondás, a kezdeti kifejezés az ág végén található behelyettesítéssel igazzá tehető
    - Pl: p, ¬q esetén: p igaz, q hamis a behelyettesítés
- A fa nem ellentmondásos ágai jelölik ki, hol igaz a kifejezés
  - Modellellenőrzésben: Ellenpéldát keresünk
  - Ezért a negált kifejezést bontjuk fel
    - A negált kifejezésen sikeres ág ellenpéldát ad
    - Ha minden ág ellentmondásos, akkor az eredeti kifejezés igaz!

Egy példa (Boole logika)

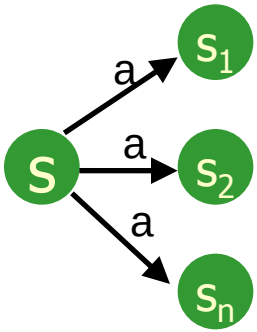
- Vizsgálandó kifejezés: $\neg ((X \wedge Y) \rightarrow (X \vee Y))$
- Az implikáció kifejtése: $\neg (\neg(X \wedge Y) \vee (X \vee Y))$
- Negált normál forma: $(X \wedge Y) \wedge \neg(X \vee Y)$
  $(X \wedge Y) \wedge (\neg X \wedge \neg Y)$
- Tabló konstruálás:

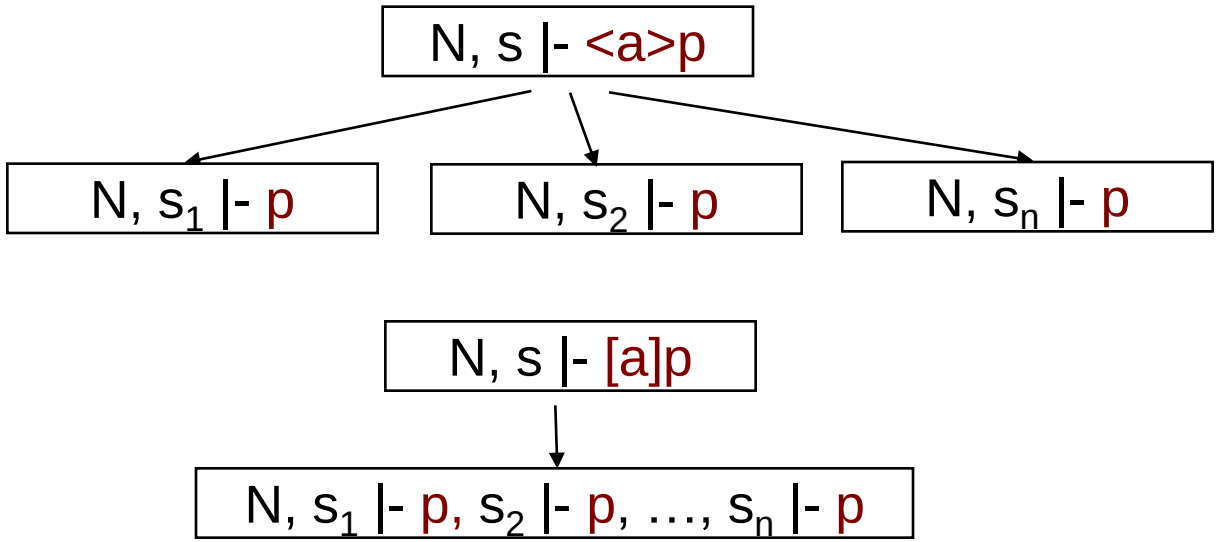


- Itt minden ág ellentmondásos
  - A kiindulásként használt kifejezés nem lehet igaz

Modellellenőrzés HML esetén

- Felbontási szabályok HML temporális operátorok esetén:



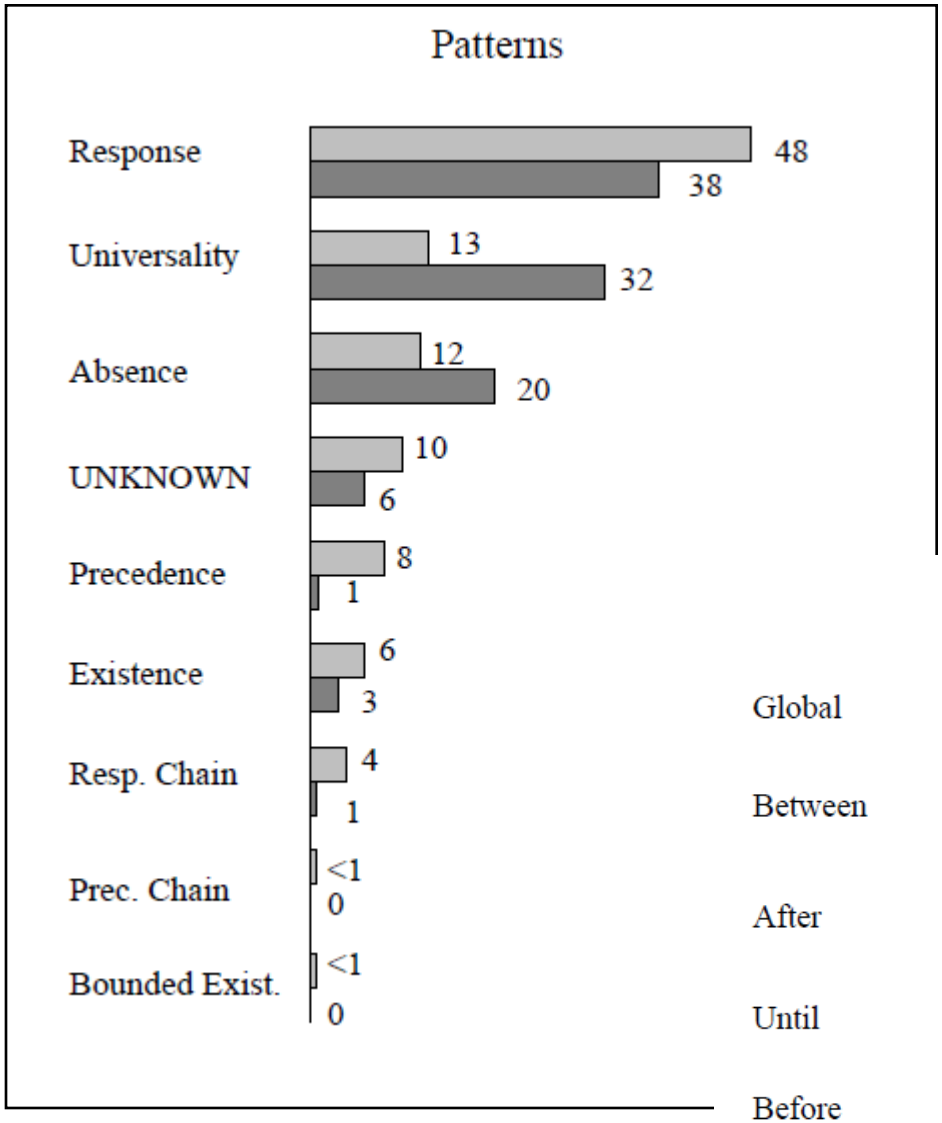


ahol $\{s_1,s_2,..,s_n\}=\{s' \mid s \rightarrow^a s'\}$

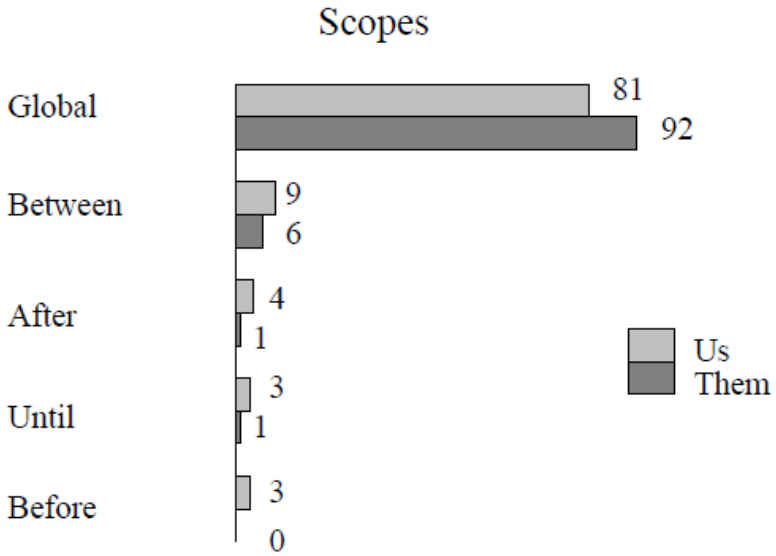
- A modell állapottere alapján történik a felbontás!
- Sikeres ág: s |- true vagy s |- [a]p, ahol nincs a átmenet
- Ha a bemenet negált kifejezés: A sikeres ág ellenpélda

Tartalomjegyzék

Biztonsági követelmények formalizálása

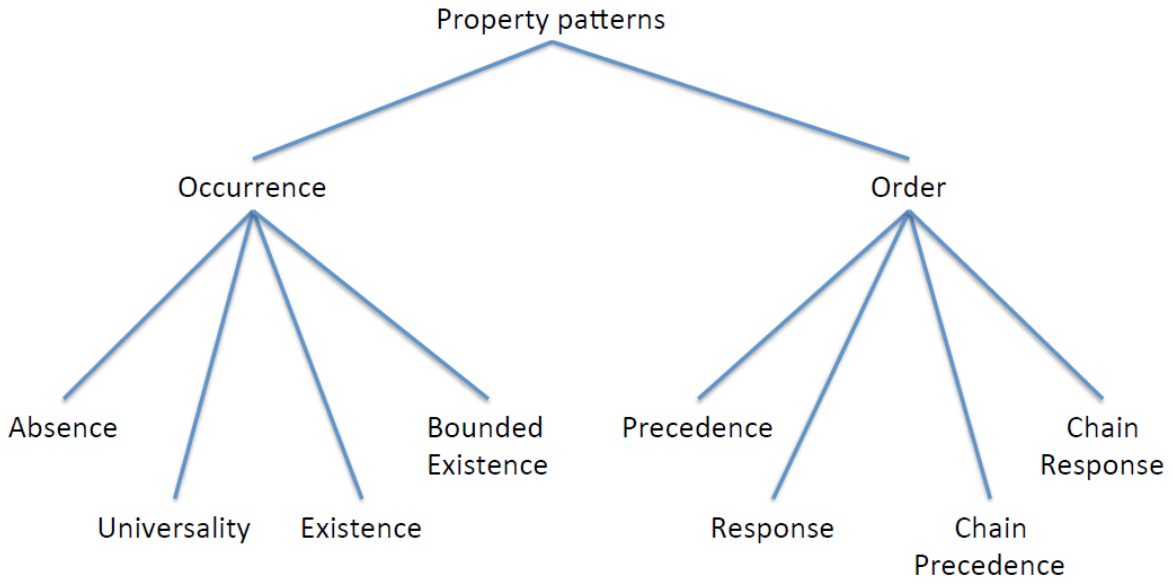


A biztonsági követelmények jelentős hányada meghatározott mintákra illeszkedik

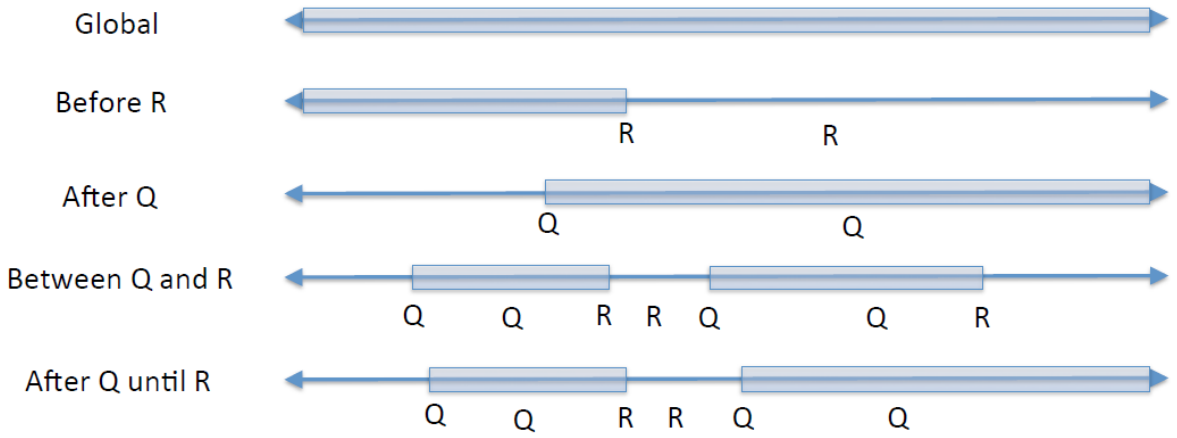


Követelmény minták

Minták (sorrendi előírás):



Hatókörök (további kijelentésekhez képest):



Példák a minták megjelenésére

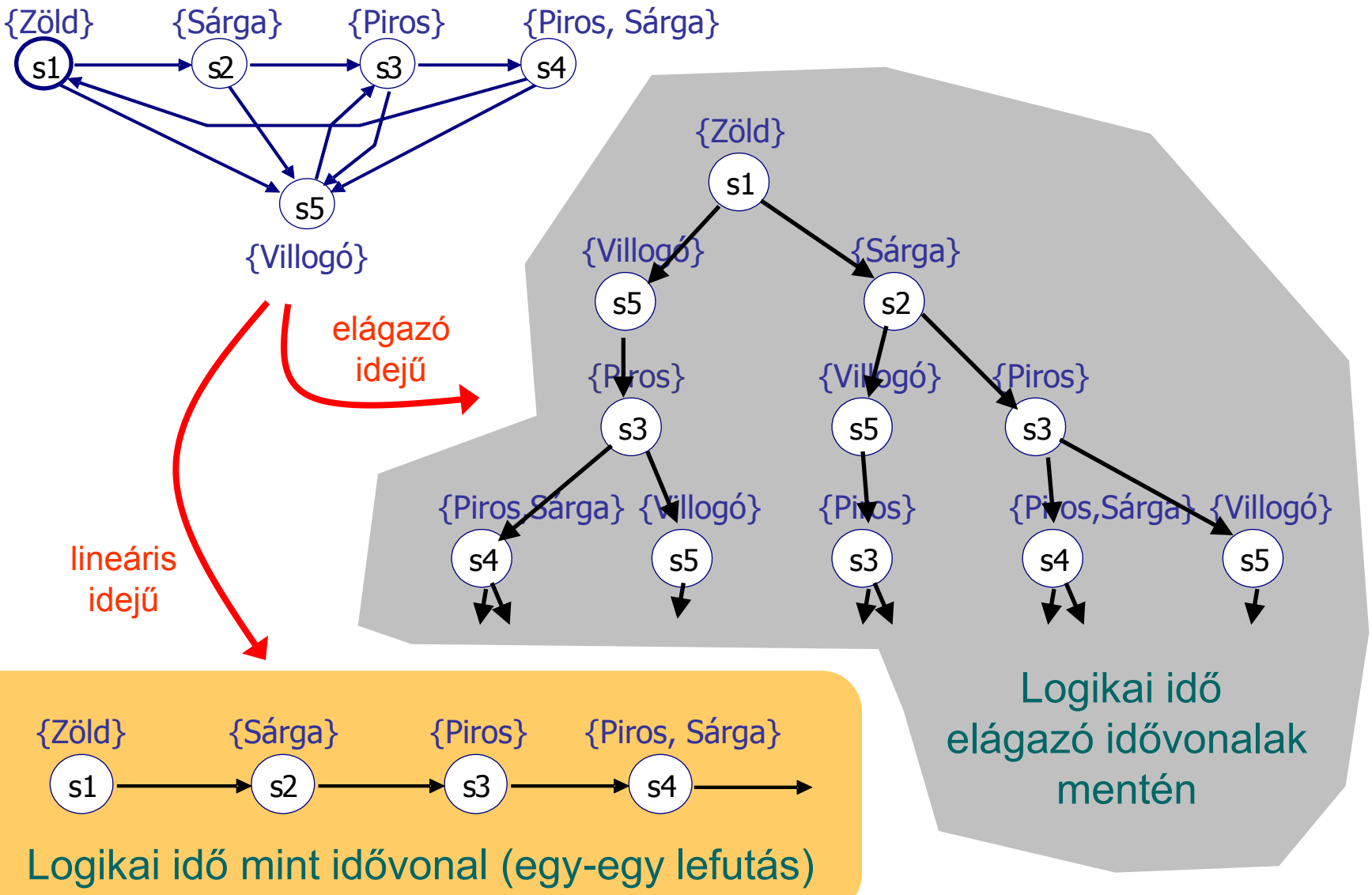
- Response minta Global hatókörben:
  - A végrehajtás során bármikor
    ha Request kérés következik be, akkor
    vagy egy Reply vagy egy Reject válasznak kell
    következnie
- Precedence minta After hatókörben:
  - A NormalMode állapot bekövetkezése után
    a ResourceGranted állapotot
    meg kell előznie a ResourceRequest állapotnak

Előzetes: Követelmény minták formalizálása

Universality within scope	**Property in LTL**	**Meaning of LTL operators**
Event P occurs in each step of the execution globally.	G P	Globally P
Event P occurs in each step of the execution before event Q.	F Q $\rightarrow$ (P U Q)	Eventually Q implies (P Until Q)
Event P occurs in each step of the execution after event Q.	G (Q $\rightarrow$ G P)	Globally (Q implies Globally P)
Event P occurs in each step of the execution between events Q and R.	G ((Q $\wedge \neg$R $\wedge$ F R) $\rightarrow$ (P U R))	Globally ((Q and not R and Eventually R) implies (P Until R))

Existence within scope	**Property in LTL**
Event P occurs in the execution globally.	F P
Event P occurs in the execution before event Q.	$\neg$ Q W (P $\wedge \neg$ Q)
Event P occurs in the execution after event Q.	G ($\neg$Q) $\vee$ F (Q $\wedge$ F P))
Event P occurs in the execution between events Q and R.	G (((Q $\wedge \neg$R) $\wedge$ (F R)) $\rightarrow$ ($\neg$R W (P $\wedge \neg$R)))

Lineáris és elágazó idejű temporális logikák

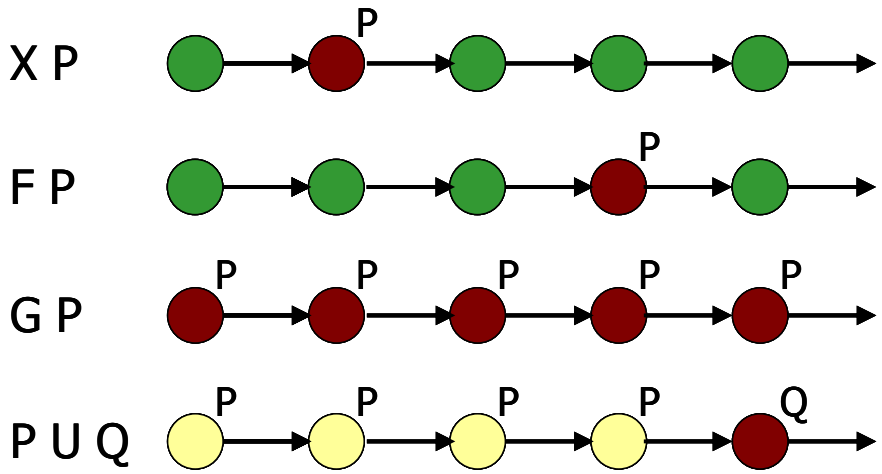


PLTL: Egy lineáris idejű temporális logika

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: X, F, G, U informálisan:



PLTL formális szintaxis

Az érvényes PLTL kifejezések halmaza a következő szabályokkal képezhető:

- **L1**: Minden P atomi kijelentés egy kifejezés.
- **L2**: Ha p és q egy-egy kifejezés, akkor $p \wedge q$ illetve $\neg p$ is
- **L3**: Ha p és q egy-egy kifejezés, akkor p U q illetve X p is

BNF: $PLTL ::= P \mid \neg p \mid p \wedge p \mid X\ p \mid p\ U\ p$

Operátorok precedenciája növekvő sorrendben:

$\equiv, \Rightarrow, \vee, \wedge, \neg$, (X,U)

„Kifejezhető” temporális operátorok:

F p jelentése true U p

G p jelentése $\neg F(\neg p)$

p WB q jelentése $\neg((\neg p)\ U\ q)$

p B q jelentése $\neg((\neg p)\ U\ q) \wedge F\ q$

PLTL szemantika

Jelölések:

- M=(S, R, L) Kripke-struktúra
- $\pi=(s_0, s_1, s_2,\ldots)$ az M egy útvonala, ahol
  s_0 a kezdőállapot és $\forall i\geq 0: (s_i, s_{i+1})\in R$
- $\pi^i=(s_i, s_{i+1}, s_{i+2},\ldots)$ a π útvonal szuffixe i-től.
- $M,\pi \models p$ jelentése: az M modellben a π útvonalon igaz p.

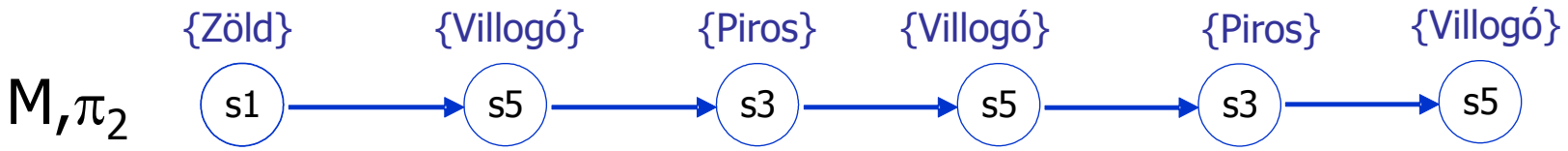
Formális szemantika a szintaxis alapján képzett kifejezésekhez:

- **L1**: $M,\pi \models P$ a.cs.a. $P\in L(s_0)$
- **L2**: $M,\pi \models p\wedge q$ a.cs.a. $M,\pi \models p$ és $M,\pi \models q$
  $M,\pi \models \neg q$ a.cs.a. $M,\pi \models q$ nem igaz.
- **L3**: $M,\pi \models (p\ U\ q)$ a.cs.a.

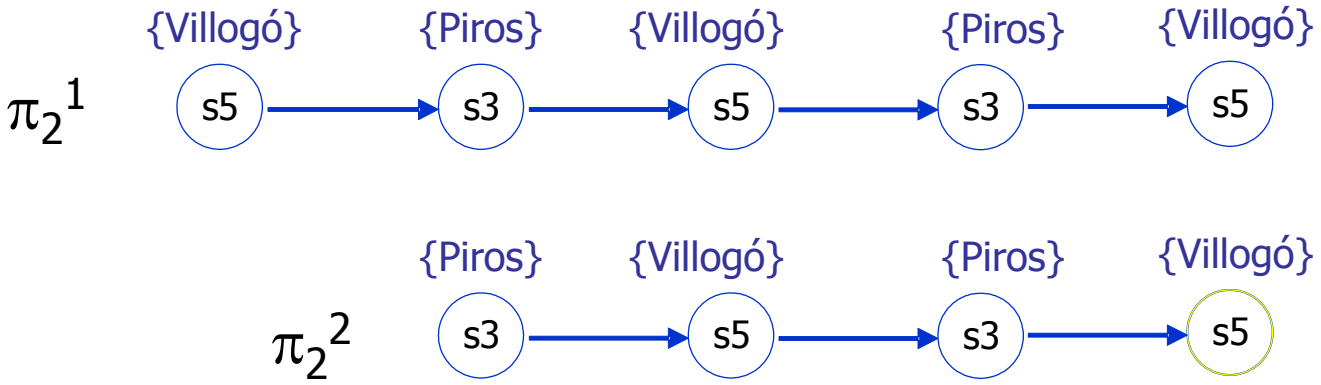
$$\exists j \geq 0 : (\pi^j \models q \text{ valamint } \forall 0 \leq k < j : \pi^k \models p)$$

$M,\pi \models X\ p$ a.cs.a. $\pi^1 \models p$

Egy példa



- $M,\pi_2 \models F$ (Villogó $\Rightarrow$ X Piros),
  mert van olyan szuffix, hogy Villogó $\Rightarrow$ X Piros igaz



Példa: Klímaberendezés működése

AP={Kikapcsolva, Bekapcsolva, Elromlott, GyengénHűt, ErősenHűt, Fűt, Szellőztet}

- A klímát be lehet kapcsolni (csak egyszer?):
  F (Bekapcsolva)
- A klíma előbb-utóbb mindig elromlik:
  G F (Elromlott)
- Ha a klíma elromlik, mindig megjavítják:
  G (Elromlott $\Rightarrow$ F ($\neg$Elromlott))
- Ha a klíma elromlott, nem fűthet:
  G ($\neg$(Elromlott $\wedge$ Fűt))

Példa: Klímaberendezés (folytatás)

AP={Kikapcsolva, Bekapcsolva, Elromlott, GyengénHűt, ErősenHűt, Fűt, Szellőztet}

- A klíma csak úgy romolhat el, ha be volt kapcsolva:

  $G\ (\ X\ \text{Elromlott} \Rightarrow \text{Bekapcsolva})$
- A fűtési fázis befejezésekor szellőztetni kell:

  $G\ ((\text{Fűt} \wedge X(\neg \text{Fűt})) \Rightarrow X\ (\text{Szellőztet}))$

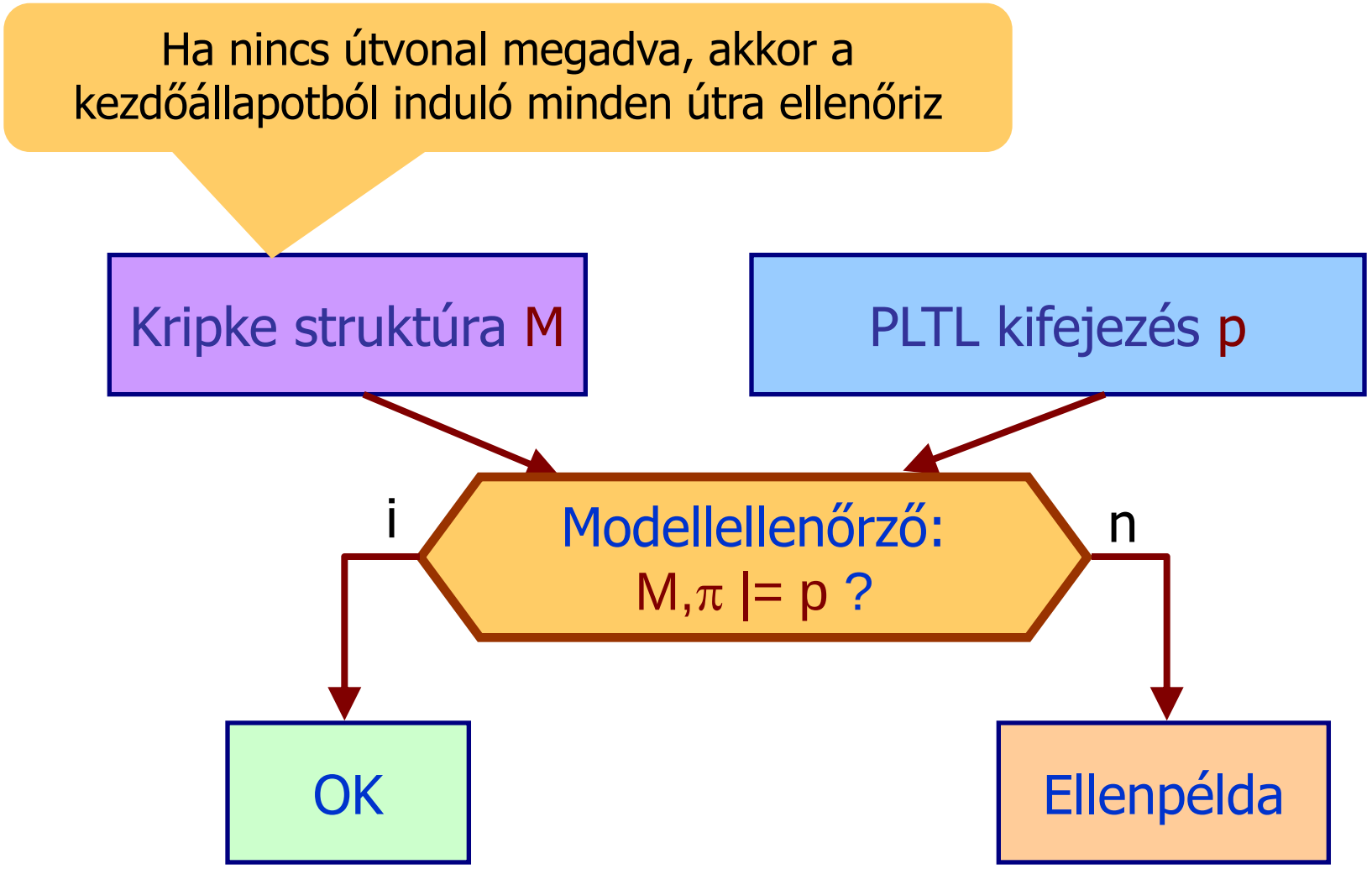
  de el is romolhat:

  $G\ ((\text{Fűt} \wedge X(\neg \text{Fűt})) \Rightarrow X\ (\text{Szellőztet} \vee \text{Elromlott}))$
- Szellőztetés után mindaddig nem hűthet erősen, míg egy gyenge hűtéssel nem próbálkozott:

  $G\ ((\text{Szellőztet} \wedge X(\neg \text{Szellőztet})) \Rightarrow X(\neg \text{ErősenHűt}\ U\ \text{GyengénHűt}))$

Tartalomjegyzék

A modellellenőrzés feladata

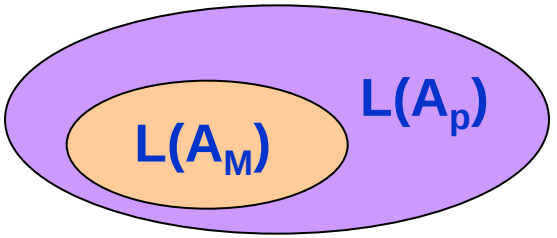


Az automata alapú megközelítés

- Betűk a 2^{AP} ábécéből
  - A KS egy s állapotához L(s) betű tartozik
  - Pl. {Piros, Sárga} az ábécé egy betűje a példában
- 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.

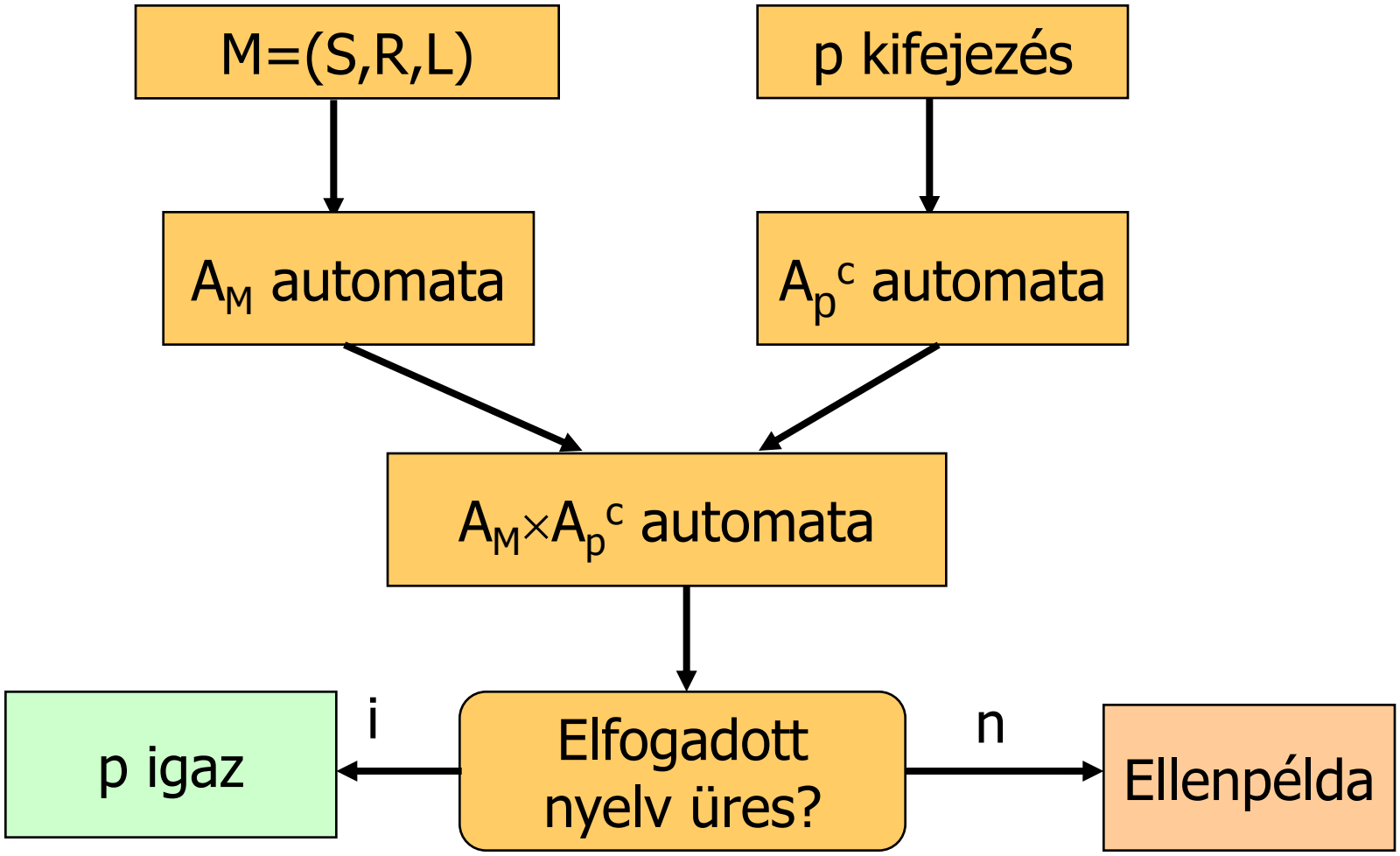
Modellellenőrzés az automatákkal

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



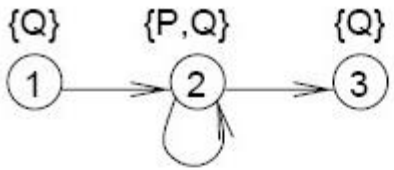
- A kérdés átalakítása:
  - Nyelvek metszetének üressége: $L(A_M) \cap L(A_p)^c = 0$, itt $L(A_p)^c$ a komplementer nyelv
  - A „modell automata” és a „komplementer tulajdonság automata” $A_M \times A_p^c$ szinkron szorzatát képezve, az általa elfogadott nyelv üres-e?
    - Ha üres, akkor $M,\pi \models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

Az automata alapú modellellenőrzés

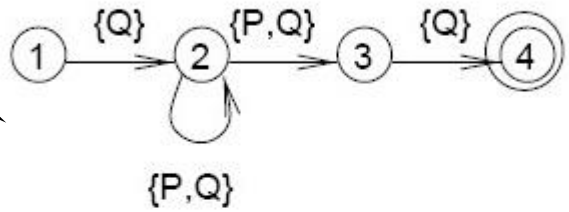


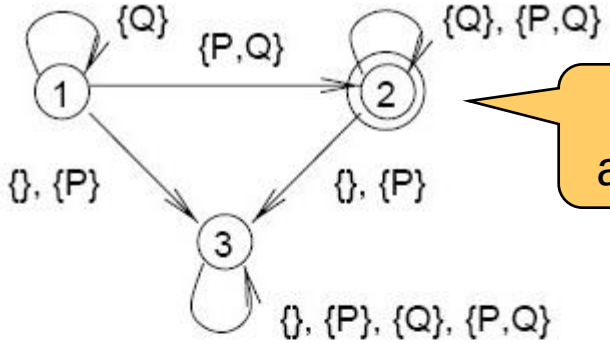
Példa: F P ∧ G Q ellenőrzése

M modell

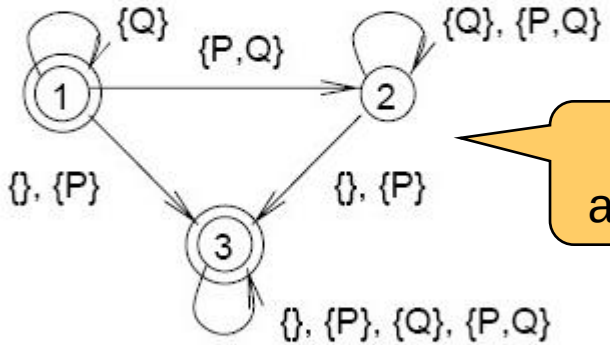


A_M automata



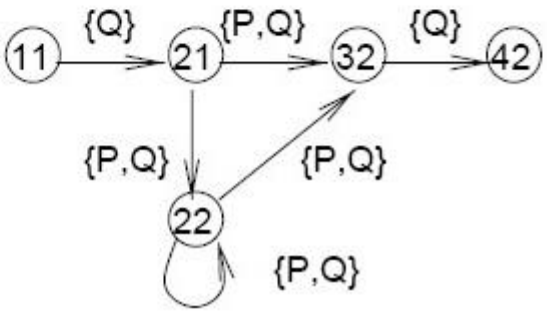


A_p automata



A_p^c automata

A_M és A_p^c automaták szinkron szorzata



Nincs elfogadó állapot: Nincs ellenpélda

„On-the-fly” modellellenőrzés

- 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 konkurens, magasabb szintű modellből való származtatás esetén)
  - Modell és tulajdonság automata szimultán előállítása

A_M konstruálása M alapján

- Címkék áthelyezése a kimenő élekre
- Elfogadó állapot konstruálása és behúzása a végállapotoktól (ahonnan nincs kimenő él)
- Az automata:

$$A_M=(2^{AP}, S\cup\{s_f\}, \{s_0\}, \rho, \{s_f\})$$

ahol az állapotátmeneti reláció:

$$\rho=\{ (s,L(s),t) \mid (s,t)\in R \} \cup$$
$$\{(s,L(s),s_f) \mid \text{nincs t, hogy } (s,t)\in R \}$$

A_p konstruálása (1)

- A_p automata: csak olyan utak, amelyeken p teljesül
- Ötlet: A tabló módszerrel kifejteni a kifejezéseket,
  és ennek során meghatározni A_p állapotait és átmeneteit
  - Meghatározni, milyen címkézés kell legyen egy adott állapotban (az aktuális állapotra vonatkozó, Boole-logikai felbontás alapján)
  - Meghatározni, mi vonatkozik a következő állapotból induló szuffixre (a rákövetkező állapotokra vonatkozó részkifejezések meghatározása a temporális operátorok felbontása alapján)
  - A következő állapotokban folytatni az ottani kifejezések felbontását
- Először: Negált normál formára hozás
  - Boole operátorokra: ld. de Morgan szabályok
  - Temporális operátorokra:

    $\neg(X\ p) = X\ (\neg p)$

    $\neg(p\ U\ q) = (\neg p)\ R\ (\neg q)$ ahol R a „release” operátor:

    $p\ R\ q = q \wedge (p \vee X\ (p\ R\ q))$

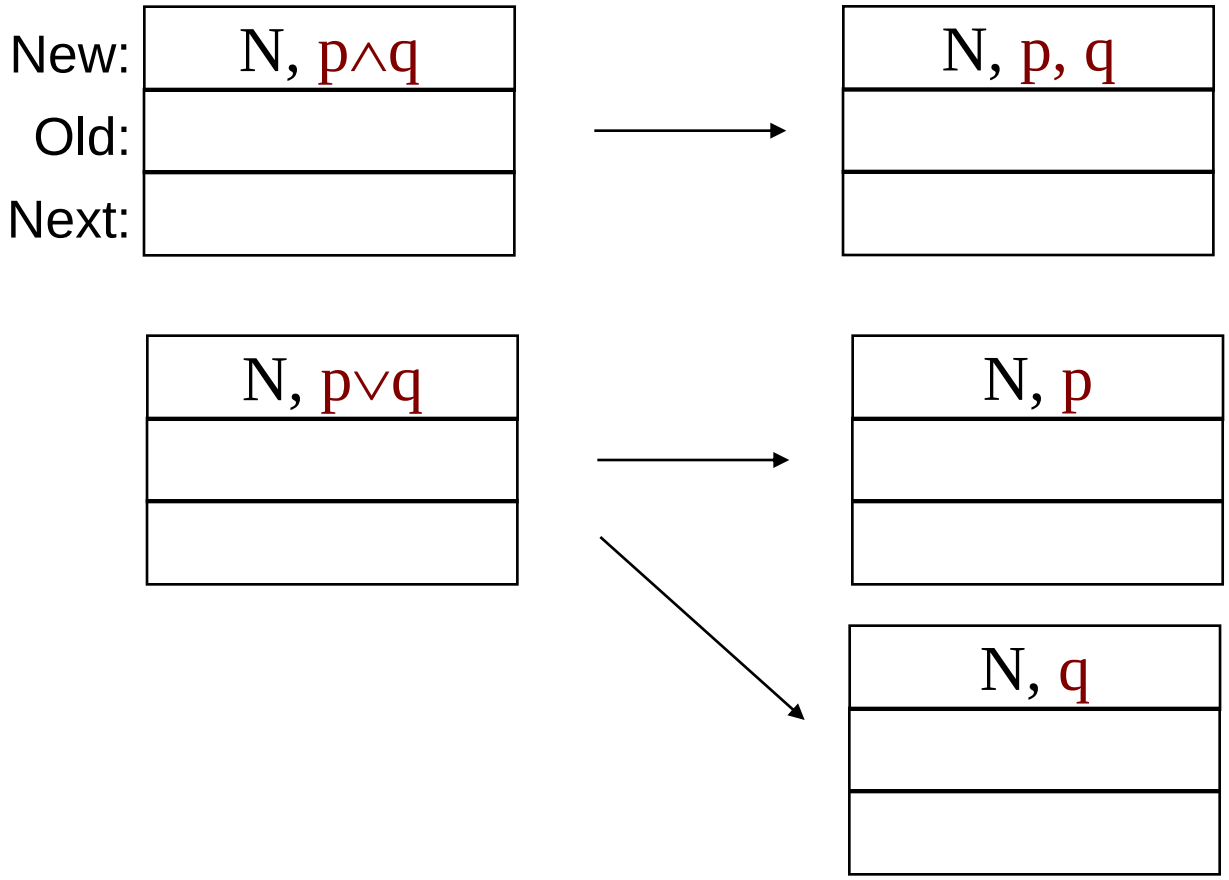
A_p konstruálása (2)

- A tablóhoz használt adatstruktúra (rekord):
  - New: aktuális felbontandó kifejezés (kifejezéslista)
  - Old: felbontott kifejezés (állapot címkét határozza meg)
  - Next: következő állapotra vonatkozó kifejezés (átmeneteket határozza meg)

New:	N, p
Old:	
Next:	

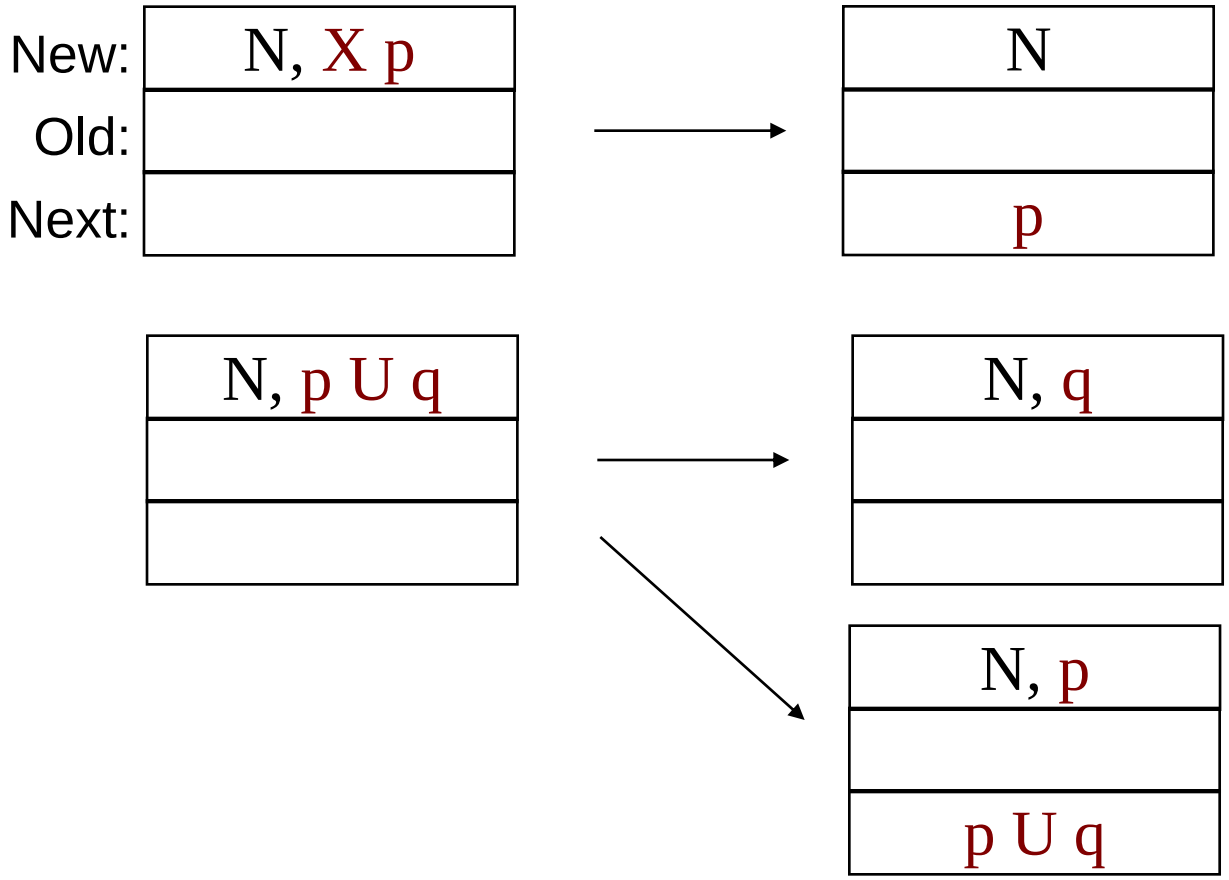
A_p konstruálása (3)

- A felbontási szabályok $\wedge$ és $\vee$ esetén:



A_p konstruálása (4)

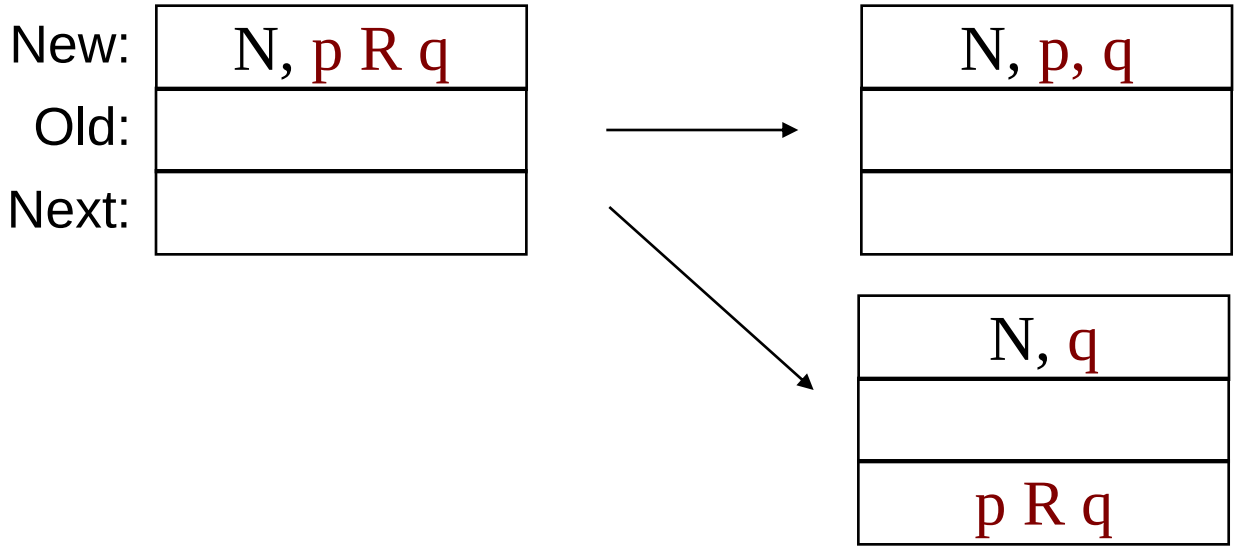
- A felbontási szabályok X és U esetén:



mivel $p\ U\ q = q \vee (p \wedge X(p\ U\ q))$

A_p konstruálása (5)

- A felbontási szabályok R esetén:

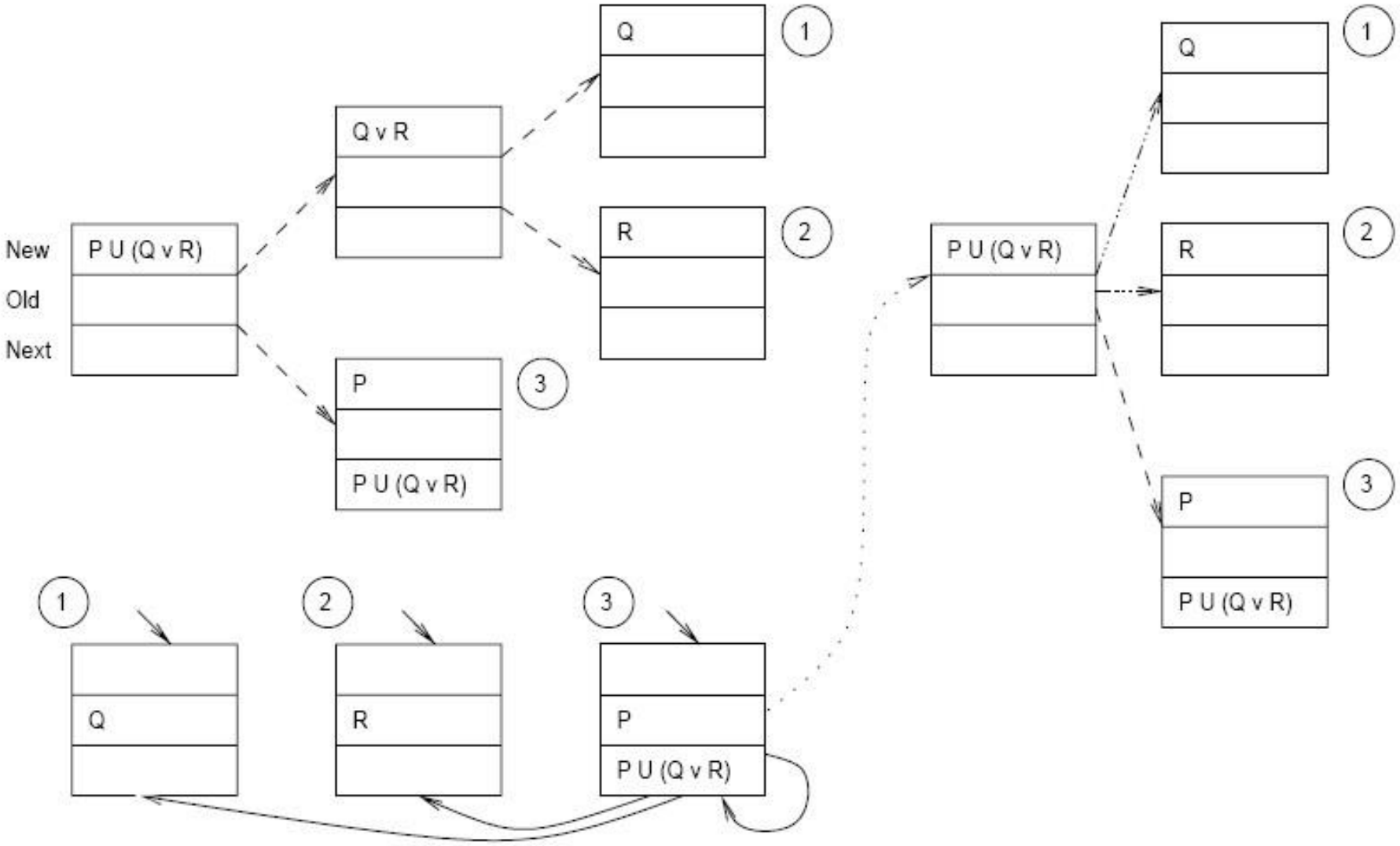


mivel $p\ R\ q = q \wedge (p \vee X(p\ R\ q))$

A_p konstruálása (6)

- A tabló egy csomópontjából az A_p automata egy állapota lesz, ha:
  - a csomópont New mezőjében csak atomi kijelentések maradnak (ezeket átírjuk az Old mezőbe a címkézés meghatározásához),
  - és A_p -nek még nincs olyan felvett állapota, aminek Old mezőjében ugyanilyen kijelentések már vannak (egyébként ugyanazt az állapotot kaptuk)
- A_p egy s állapotának létrehozásakor:
  - A Next mezőben lévő kifejezés alapján új felbontás indítása, a Next mező New-ba való átírásával (következő állapotra vonatkozott!)
  - A Next mezőben lévő kifejezés alapján indított új felbontással előálló új állapotokhoz átmenetet kell majd húzni s-ből
- Összegezve:
  - A_p állapotok létrejönnek, ha New tovább nem bontható
  - A_p átmenete létrejön a Next mező felbontása alapján létrejövő állapotokhoz (következő állapotot határozza meg)

Példa: P U (Q ∨ R)

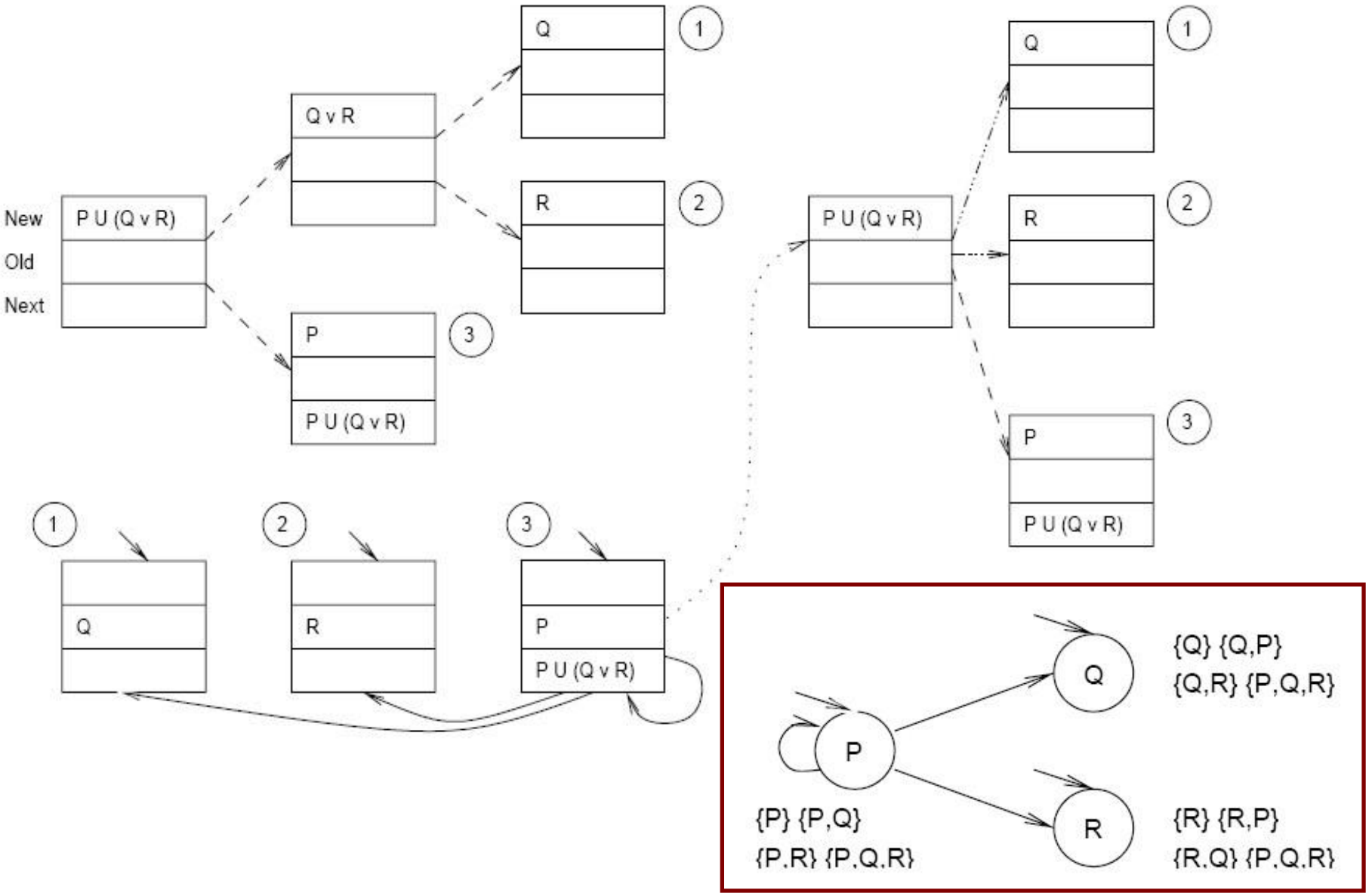

New
Old
Next
P U (Q v R)
Q v R
Q
R
P
P U (Q v R)
1
2
3
Q
R
P
P U (Q v R)
1
2
3
P U (Q v R)
Q
R
P
P U (Q v R)
1
2
3

A_p konstruálása (7)

- További elemek A_p-ben:
  - Kezdőállapot(ok):
    - Az első felbontás alapján
  - Elfogadó állapotok:
    - Ahol a Next mező üres volt (nincs további feltétel)
  - Állapotok címkézése: Az AP olyan részhalmazai, amelyek kompatibilisek az Old mezőben lévő kijelentés-halmazzal:
    - Minden atomi kijelentés benne, ami az Old-ban is
    - Nincs olyan atomi kijelentés benne, aminek negáltja van Old-ban

  mivel minden megengedett viselkedést le kell írni!

Példa: P U (Q ∨ R)


New
Old
Next
P U (Q v R)
Q v R
Q
R
P
P U (Q v R)
1
2
3
Q
R
P
P U (Q v R)
1
2
3
P U (Q v R)
Q
R
P
P U (Q v R)
1
2
3
Q
R
P
{Q} {Q,P}
{Q,R} {P,Q,R}
{P} {P,Q}
{P,R} {P,Q,R}
{R} {R,P}
{R,Q} {P,Q,R}

Tartalomjegyzék

A PLTL modellellenőrzés komplexitása

- M=(S,R,L) struktúrán a p kifejezés modell-ellenőrzésének worst case időkomplexitása: $O(|S|^2 \times 2^{|p|})$
  - Itt |S| az állapotok száma, |p| az operátorok száma
  - $|S|^2$ a modell automata átmeneteinek száma (max. négyzetes, átlagos esetben csak lineáris)
  - $2^{|p|}$ a tulajdonság automata átmeneteinek száma: a tabló során ennyi a rész-kifejezések (mint lehetséges felbontandó kifejezések) maximális száma
    - Az exponenciális komplexitás riasztó, de
      - általában a PLTL kifejezés rövid,
      - a komplexitás S mérete miatt lesz nagy
  - $O(|S|^2 \times 2^{|p|})$ a szinkron szorzat automata állapotterének méretéből adódik (elfogadó állapot keresése)

A SPIN modellellenőrző