Logikák véges fákon

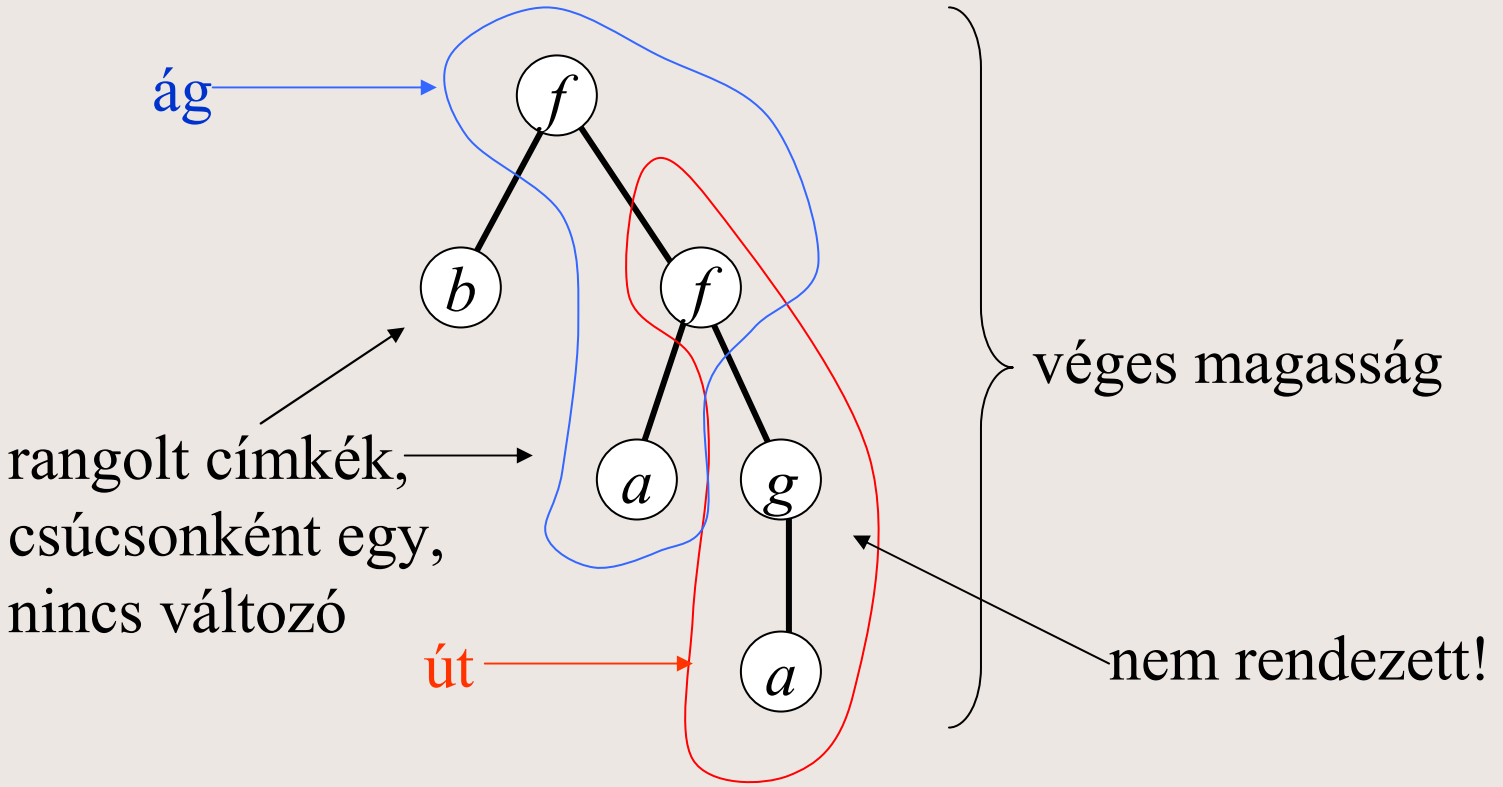
Iván Szabolcs
Szegedi Tudományegyetem
Számítástudomány Alapjai Tanszék

Tartalom

- FO[<], MPL és CTL*
  - Hafer, Thomas (1987)
  - Moller, Rabinovich (1999-2003)
- CTL: TL[EX], TL[EF], TL[EX+EF]
  - Bojanczyk, Walukiewicz (2004)

Fák

- Fáink ezúttal:



MPL emlékeztető

- MPL: Monadic Path Logic

$\{x, y, z, x_1, \ldots\}$: csúcsértékű változók

$\{X, Y, Z, X_2, \ldots\}$: ágértékű változók

Szintaxis

- $\mathrm{F} \rightarrow P_a(x) \mid x <_{ch} y \mid x = y \mid \neg\mathrm{F} \mid \mathrm{F} \wedge \mathrm{F}$
- $\mathrm{F} \rightarrow \exists x.\mathrm{F} \mid \exists X.\mathrm{F} \mid x \in X$

MPL szemantika

$P_a(x)$: x címkéje a

$x <_{\mathrm{ch}} y$: x-nek fia y

$=$, $\wedge$, $\neg$: szokásos

$x \in X$: az x csúcs rajta van az X *ágon*

$\exists x.\mathrm{F}$: van olyan x csúcs, amire F

$\exists X.\mathrm{F}$: van olyan X *ág*, amire F

Rövidítések: $\forall$, $\vee$, $\rightarrow$, stb.

MPL példa

- „Van *két* olyan ág, amin a csúcsok végig *f*-fel, *g*-vel vagy *a*-val vannak címkézve”

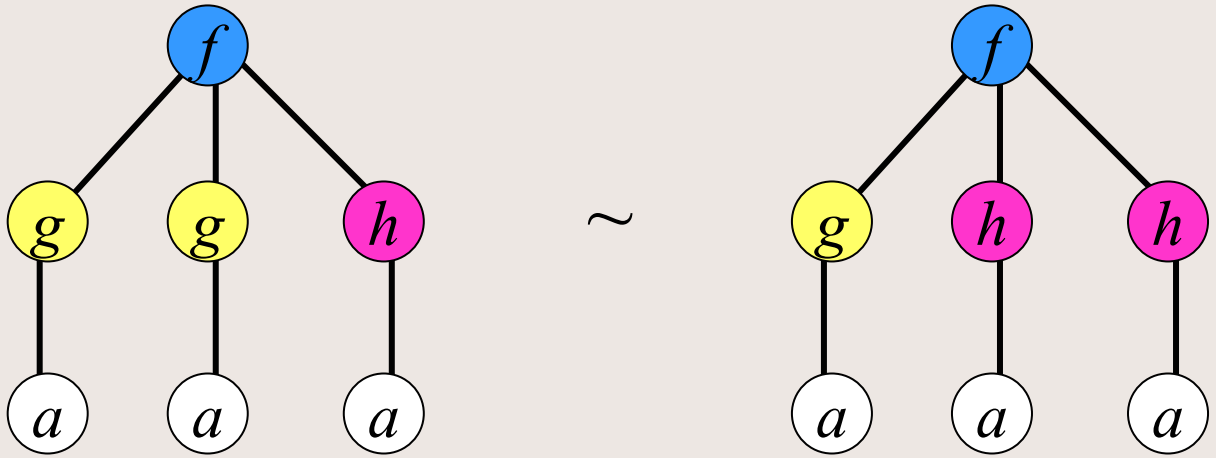
$$\exists X \exists Y \;\; \forall x \; x \in X \rightarrow P_f(x) \vee P_g(x) \vee P_a(x) \;\; \wedge$$
$$\forall x \; x \in Y \rightarrow P_f(x) \vee P_g(x) \vee P_a(x) \;\; \wedge$$
$$\exists x \; x \in X \wedge x \notin Y$$

Biszimuláció

- Legyen t_1 és t_2 két fa. $V(t_1) \times V(t_2)$ egy R részhalmaza (reláció) *biszimuláció*, ha $root(t_1)Rroot(t_2)$, továbbá ha uRv fennáll, úgy
  - u és v címkéje megegyezik;
  - u bármelyik u' gyerekére van olyan v' gyereke v-nek, amire $u'Rv'$;
  - és szimmetrikusan is.

Biszimuláció példa

Biszimuláció-ekvivalens fák



Azaz MPL-ben *nem* csak biszimuláció-invariáns tulajdonságok fejezhetők ki

CTL*

- Változómentes logika
  - Így csak legfeljebb unáris lekérdezések definiálhatók benne

Szintaxis

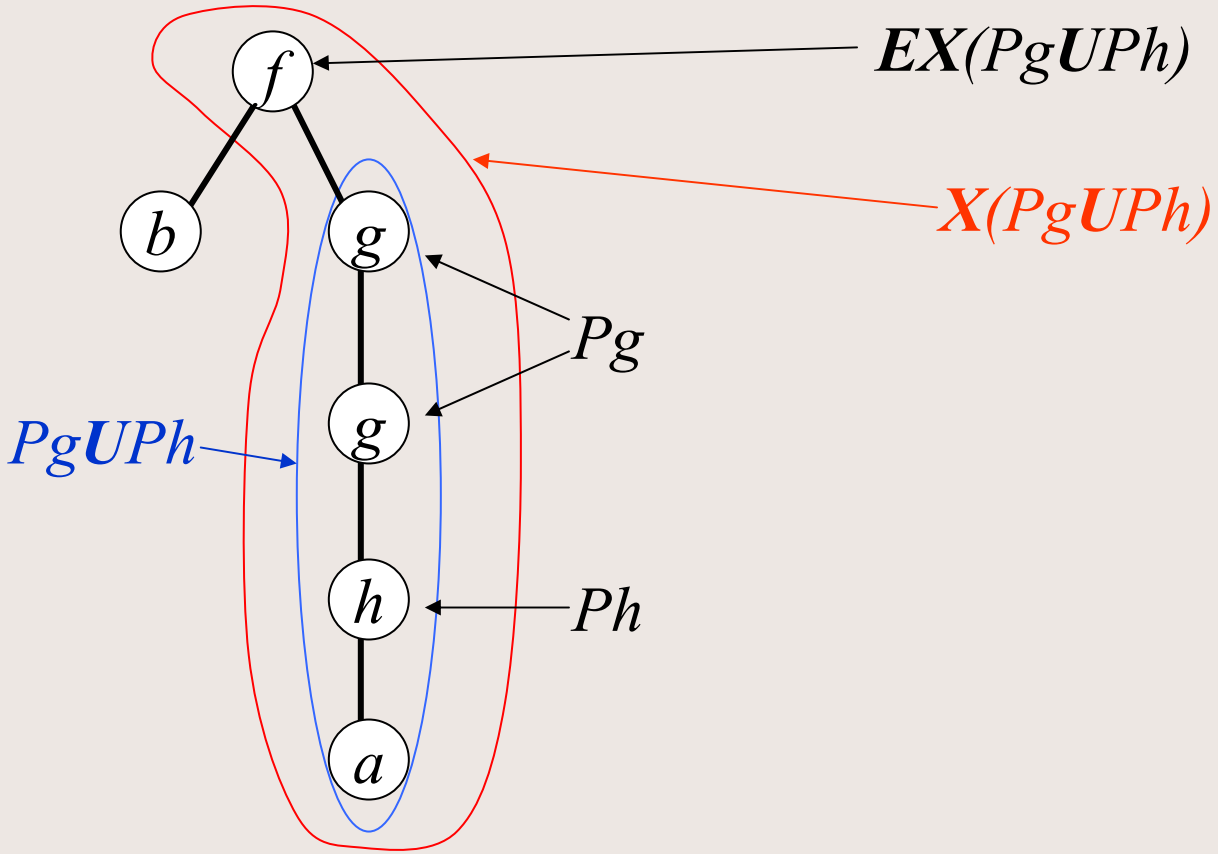
- $\mathrm{F} \rightarrow P_a \mid \neg \mathrm{F} \mid \mathrm{F} \wedge \mathrm{F} \mid \boldsymbol{E}\mathrm{G}$
- $\mathrm{G} \rightarrow \mathrm{F} \mid \neg \mathrm{G} \mid \mathrm{G} \wedge \mathrm{G} \mid \boldsymbol{X}\mathrm{G} \mid \mathrm{G}\boldsymbol{U}\mathrm{G}$

F: állapotformulák G: útformulák

CTL* szemantika

- Állapotformulák egy-egy *csúcsban* lehetnek igazak vagy hamisak
  - ***E***G: van az adott csúcsból induló út, melyre G igaz
- Útformulák egy-egy útra lehetnek azok
  - Állapotformula: az út *első* csúcsára igaz
  - ***X***G: elhagyva az út első csúcsát a maradékra G
  - G***U***G': G until G'

CTL* példa



CTL* és biszimuláció

- Ha $T_1 \sim T_2$, akkor minden CTL*-formula ekvivalens rajtuk
- Tehát MPL-ben és CTL*-ban *nem* ugyanazok a fanyelvek definiálhatók

$\mathrm{CTL}^* \subset \mathrm{MPL}$

- Lineáris idejű fordítás

A teljes F (állapot)formulára:

$$\forall x \, (\, \mathrm{root}(x) \rightarrow \mathrm{F}'(x) \,)$$

Állapotformulák x-re (relativizálás):

- $P_a \Rightarrow P_a(x)$
- $\neg$F, F $\wedge$ F: nincs gond
- $\boldsymbol{E}\mathrm{G} \Rightarrow \; \exists X \, (x \in X \wedge \mathrm{G}'(X, x))$

$CTL^* \subset MPL$

Útformulák X-re x-ből:

- $F \Rightarrow F'(x)$
- $\neg G$, $G \wedge G$: nincs gond
- $\boldsymbol{X}G \Rightarrow \exists y(x <_{ch} y \wedge y \in X \wedge G'(X, y))$
- $G \boldsymbol{U} H \Rightarrow$

$$\exists y(y \in X \wedge x <^{*}_{ch} y \wedge H'(X, y) \wedge$$

$$\forall z((x <^{*}_{ch} z \wedge z <^{+}_{ch} y) \rightarrow G'(X, z)))$$

MPL szemantikus fragmense

- Tehát CTL* $\subset$ MPL. Az elválasztó nyelv egy *nem* biszimuláció-invariáns nyelv volt.
- Az ilyen fanyelvek *egyike sem* definiálható CTL*-ban.
- Vizsgáljuk most MPL azon formuláinak halmazát, amik biszimuláció-invariáns nyelvet adnak meg.

MPL $\subseteq$ FO[<]

- Lineáris idejű fordítás:

  $\exists X.\mathrm{F} \quad \Rightarrow \quad \exists z_X \; leaf(z_X) \wedge \mathrm{F'}$

  $x \in X \quad \Rightarrow \quad x <^*_{\mathrm{ch}} z_X$

  Többi konnektívával nincs gond

Ezzel a konstrukcióval az X utat azonosítjuk annak z_X végpontjával.

Fontos, hogy fáink *végesek*!

FO[<] szerkezete

- Bármely $n > 0$ esetén ekvivalencia erejéig csak véges sok olyan különböző FO[<]-mondat létezik, melynek mélysége max. n.
- Mély eredményeket felhasználva kijön, hogy *széles* fákra (= minden gyerek végtelen sokszor szerepel) FO[<]=CTL*
- Ha az adott fanyelv biszimuláció-invariáns, akkor relaxálhatunk széles fákra

FO[<] és CTL*

- Összefoglalva tehát az eddigieket, egy fanyelv pontosan akkor definiálható CTL*-ban, ha biszimuláció-invariáns és definiálható FO[<]-ben.
- Kérdés: hogyan bővíthetjük ki CTL*-ot úgy, hogy megkapjuk az FO[<]-ben definiálható összes fanyelvet?

Counting-CTL*

- Szintaxis kibővül, új állapotformulák:
- $F \rightarrow \boldsymbol{D}^{n}F$

minden *n*-re (a maximális rangig)

- $\boldsymbol{D}^{n}F$ jelentése: legalább *n* különböző gyerekre fennáll F

Counting-CTL* példa

„van legalább két olyan út, ami csak az *f,g* vagy *a* címkéket használja”

ugyanaz, mint

„van egy olyan út, aminek az eleje csak az *f,g* és *a* címkéket használja, majd egyszer csak egy olyan csúcshoz ér, aminek legalább két fiából van olyan út, hogy csak az *f, g* és *a* címkéket használja”

Counting-CTL* példa

A formula tehát:

$\boldsymbol{E} \quad Pf \vee Pg \vee Pa \; \boldsymbol{U}$

$Pf \vee Pg \vee Pa \wedge \boldsymbol{D^2 E}$

$Pf \vee Pg \vee Pa \; \boldsymbol{U}$

$Pf \vee Pg \vee Pa \wedge \neg \boldsymbol{X}\text{true}$

Counting-CTL* $\subseteq$ MPL

- Csak a $\boldsymbol{D^n}$ modalitásokat kell kifejezni, de ez könnyű (n új változóra relativizálunk, mind gyereke az eredetinek).

Pl. D^2F x-re relativizálva:

$\exists x_1 \exists x_2$

$$x <_{ch} x_1 \wedge x <_{ch} x_2 \wedge x_1 \neq x_2 \wedge F'(x_1) \wedge F'(x_2)$$

MPL $\subseteq$ Counting-CTL*

- Ismét kihasználjuk, hogy rögzített maximális kvantifikációs mélység esetén ekvivalencia erejéig csak véges sok különböző MPL formula létezik
- Újabb mély bizonyítás, ezúttal széles fák nélkül

CTL* összefoglalás

- A CTL*-ban definiálható nyelvek osztálya pontosan az MPL-ben (vagy FO[<]-ben) definiálható biszimuláció-invariáns nyelvek osztályával esik egybe
- A Counting-CTL*-ban definiálható nyelvek osztálya pedig pontosan az ezen logikákban definiálható nyelvekével

CTL* megjegyzések

- Bináris fákat tekintve
  - D^0F = true
  - D^1F ugyanaz, mint ***EX***F
  - D^2F ugyanaz, mint ***AX***F
  - D^nF = false, ha n>2
- Vagyis ebben az esetben CTL^*=FO[<]

CTL* megjegyzések

- Azzal, hogy ezen logikák egybeesnek, nem kerültünk közelebb a *definiálhatóság* problémájának megoldásához
- Vagyis egyelőre nem tudjuk, hogy eldönthető-e egy adott *A* faautomatáról, hogy *L(A)* definiálható-e FO[<]-ben
- Az eszközök kifejlesztéséhez előbb szintaktikus alosztályokat vizsgálunk

CTL

- A CTL* egy széles körben alkalmazott fragmense
- Csak állapotformulái vannak:

$$\mathrm{F} \rightarrow P_a \mid \mathrm{F} \wedge \mathrm{F} \mid \neg \mathrm{F} \mid \boldsymbol{EX}\mathrm{F} \mid \boldsymbol{EU}(\mathrm{F},\mathrm{F})$$

- Szemantika ugyanaz, mint CTL* esetén
- Szintén nem megoldott a definiálhatóság

CTL származtatott modalitások

- Az ***EX*** és ***EU*** modalitásokból egyebek is képezhetőek
- ***EF*** F = ***EU***(true, F)
  - Van olyan csúcs, amire F
- ***EG*** F = ***EU***(F, F $\wedge \neg$***EX***(true))
  - Van olyan maximális út, amire végig F
- Csak ezen modalitásokat engedélyezve CTL-en belüli fragmenseket kapunk

TL[EX]

- CTL-ben csak az ***EX*** modalitást engedjük meg
- Módosíthatjuk kicsit a szemantikát, hogy rendezett fákon dolgozzunk: $\boldsymbol{EX_0}$, $\boldsymbol{EX_1}$
- Könnyen látható, hogy *n* egymásba ágyazott ***EX*** operátor a fában legfeljebb *n* mélyen lévő csúcsokat „látja”

TL[EX]

- Tehát TL[EX]-ben csak a definit fanyelvek definiálhatók
- Azokra pedig mindre lehet adni EX-formulát, vagyis pontosan a definit fanyelvek definiálhatók TL[EX]-ben
- Ez a kérdés eldönthető

TL[EF]

- ***EF*** F legyen akkor igaz, ha létezik olyan nem-gyökér csúcs, amire F teljesül
- Ez nem teljesen esik egybe az eddigi ***EF*** operátorral:
  - eddigi ***EF*** F $\Rightarrow$ F $\vee$ ***EF*** *F*
  - Az új ***EF*** F: korábban ***EXEF*** F
- TL[EF]: ***EF*** az egyetlen megengedett modalitás

TL[EF] bináris fákra

- A következőkben olyan fákon fogunk dolgozni, melyekre
  - minden nem-levél csúcsnak pontosan két fia van
  - minden címkének lehet kettő vagy nulla is az aritása
- Ez utóbbi nem gond, ***EF***-ben kifejezhető a „nem levél” állítás: ***EF*** true

Típusok

- Vegyük az L reguláris fanyelv A minimális faautomatáját; továbbiakban L adott
- A t fa *típusa* legyen t értéke A-ban
- A t fára és $a \in \Sigma$ címkére $t[a]$ legyen t, a-ra átcímkézett gyökérrel
- A t fa *kései típusa* (delayed type) egy $\Sigma \to A$ leképezés
  - $a \in \Sigma$-hoz rendeli $t[a]$ értékét A-ban

Típusok

- Egy fa kései típusát egyértelműen meghatározza két részfájának típusa
- Ha x,y kései típus, a,b címkék, akkor $dtype(x(a),y(b))$ legyen az $x(a)$ és $y(b)$ típusok által meghatározott kései típus
- Ha van olyan fa, aminek kései típusa y, és van $x(a)$ típusú részfája, annak jele $(x,a) \leq y$ (vagy, a-t elhagyva $x \leq y$).

cl(φ) és sig(*t*,φ)

- Ha φ egy ***EF***-formula, jelölje cl(φ) azt a legszűkebb formulahalmazt, melynek eleme φ, és zárt a negációra és a részformulák képzésére
- Ha *t* egy fa, φ egy ***EF***-formula, legyen sig(*t*,φ) azoknak az ***EF*** $\Psi \in$ cl(φ) formuláknak a halmaza, amik *t*-re igazak

Észrevételek

- Ha L-t definiálja φ, akkor tetszőleges t fára $\mathrm{sig}(t,\varphi)$ meghatározza t kései típusát
  - mivel ezek Boolean-kombinációi mellé már csak a gyökér címkéje szükséges a teljes típushoz
- Ha t-nek részfája t', akkor:

$$\mathrm{sig}(t',\varphi) \subseteq \mathrm{sig}(t, \varphi)$$

Következmények

- Ha a-ból elérhető b, akkor

$$dtype(a,b)=dtype(b,b).$$

- Ha a_1-ből elérhető a_2 és viszont (jelben $a_1 \approx a_2$), és $b_1 \approx b_2$, akkor

$$dtype(a_1,b_1) = dtype(a_2,b_2).$$

Typeset függés

- Legyen a *t* fa valódi részfáinak típusainak halmaza a *t* typeset-je
- Tegyük fel, hogy t_1 és t_2 ugyanazzal a typesettel rendelkezik (*L*-t még mindig φ definiálja)
- Esetvizsgálattal belátható, hogy ekkor kései típusuk ugyanaz
  - A fák magasság szerinti indukciójával

Eddig TL[EF]-ről

- Ha L definiálható $\boldsymbol{EF}$-ben, akkor egy fa typesetje meghatározza annak kései típusát.
- Ezt a tulajdonságot úgy mondjuk, hogy L typeset-függő.

A typeset-függés miatt…

- Ha L typeset-függő, akkor a kései típusok egymásból való elérhetősége részbenrendezés.
  - Tegyük fel, $x \to y \to x$
  - Ekkor létezik fák végtelen $t_0, t_1, t_2, \ldots$ sorozata, t_i mindig valódi részfája t_{i+1}-nek, kései típusok $x,y,x,y,\ldots$
  - A typeset egyre bővül, de véges
  - Emiatt ekkor $x=y$ kell fennálljon

A typeset-függés miatt…

- Az *x* kései típusra nézve az *a* betű *egység*, ha *dtype(x(a),x(a))=x*
- Ha *L* typeset-függő, *y* egy kései típus, és *b,b'* egységek *y*-ra nézve, akkor

  dtype(x(a),y(b))=dtype(x(a),y(b')).

  – az előzőhöz hasonló iterációt használva

A typeset-függés miatt…

- Ha L typeset-függő, akkor $(x,a) \leq y$ maga után vonja

$$dtype(x(a),y(c))=dtype(y(c),y(c))$$

teljesülését is

- Mert $x(a)$ typesetje benne van $y(c)$-ében
- Továbbá a typeset-függés a kommutativitást is maga után vonja.

Eddig TL[EF]-ről

- Ha *L* ***EF***-definiálható, akkor typeset-függő.
- Ha *L* typeset-függő, teljesül rá az előző négy tulajdonság.
  - Kései típusok részbenrendezettek
  - Egységbetűket szabad cserélni
  - Elérhetőség elnyelő hatású a kései típusok számítására
  - Kommutativitás

Az előző négy tulajdonság

- Ha *L* bír ezen tulajdonságokkal, akkor *L* *EF-közelíthető*.
- Ha *L* EF-közelíthető, akkor (a részbenrendezés és az elnyelés miatt) ha *t* kései típusa *x*, akkor minden *x* kései típusú valódi részfájának gyökerében *x*-egység kell szerepeljen

EF-definiálás

- Az EF-közelíthető nyelvek esetében le tudjuk definiálni sorban a kései típusokat
- Vagyis ha a kései típusok egy topologikus rendezése $x_1<x_2<\ldots<x_n$, akkor konstruálhatóak ilyen sorrendben $F_1,F_2,\ldots,F_n$ formulák úgy, hogy t-re pontosan akkor igaz F_i, ha kései típusa x_i

TL[EF] vége

- Összegezve, az alábbiak ekvivalensek:
  - *L(A)* definiálható ***EF***-ben
  - *L(A)* typeset-függő
  - *L(A)* ***EF***-közelíthető
- Mivel a kései típusok meghatározhatók, így az EF-közelíthetőség (és így az EF-definiálhatóság) eldönthető.

TL[EF] kommentár

- Ami sokat számít az EF-definiálhatóság megoldásában, az az, hogy EF nagyon korrelál *A* elemeinek egymásból való elérhetőségével
- A többi modalitás esetén ez nem tűnik továbbvihetőnek, túl ***EF***-specifikus

TL[EF+EX]

- EF és EX együttes használata megengedett
- EX: egy mélységig elég nézni a címkéket
  - Definit fanyelvek
- EF: elég tudni a fában előforduló típusokat
  - Typeset-függő fanyelvek
- A kettő együttes alkalmazása most valóban a két elem egyfajta kombinációja lesz

Komponensek

- Az *A* (minimális) faautomata egy *komponense* a típusgráfjának egy erősen összefüggő komponense

Komponens-feloldhatóság

- Az *A* faautomata komponens-feloldható, ha létezik olyan *k*, melyre a következők egyértelműen meghatározzák bármely *t* fa típusát:
  - *t* típusának komponense
  - *t* kisebb, mint *k* mélyen lévő címkéi
  - *t* pontosan *k* mélyen lévő azon csúcsainak típusa, melyekre ez a típus nem *t* komponensében van

A kapcsolat

- A következők ekvivalensek:
  - *L* definiálható TL[EX+EF]-ben
  - *A(L)* komponens-feloldható
- A komponens-feloldhatóság is eldönthető tulajdonság.
  - „multicontextek” használatával
  - az eddigiekhez hasonló iterációval

Irodalom

- M. Bojanczyk, I. Walukiewicz:*Characterizing EX and EF tree logics* Lecture Notes in Computer Science, vol. 3170, p. 131-145, 2004.
- L. Libkin: *Logics for unranked trees: an overview* Lecture Notes in Computer Science, vol. 3580, p. 35-50, 2005.
- F. Moller, A. Rabinovich: *On the expressive power of CTL** Logic in Computer Science 1999, p. 360-369.
- F. Moller, A. Rabinovich: *Counting on CTL*: on the expressive power of monadic path logic* Information and Computation, vol. 184, p. 147-159, 2003.
- A. Rabinovich: *Expressive Power of Temporal Logics* Lecture Notes in Computer Science, vol. 2421, p. 57-75, 2002.