Oblikovanje programske podrške

ak.god. 2014./2015.

Temelji formalne verifikacije
Formalna verifikacija

Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikroel., računalne i inteligentne sustave

Sadržaj

- Osnove modalne i vremenske (temporalna) logike
- Definicije Kripke struktura
- Vremenska logika s grananjem
- Specifikacija ponašanja sustava CTL formalizmom

Sadržaj

- Formalna matematička logika
  - propozicijska
  - predikatna
- Preslikavanje formula predikatne logike u normalizirane klauzule
- Postupci formalne verifikacije računalnih sustava

Modalna logika

- *Modalna logika* - proširenje klasične logike "modalitetima" istinitosti (subjektivnim konceptima), kao što su npr. "što mora biti istinito" i "što može biti istinito”.
- Npr.: p = atomički propozicijski simbol

Neka je: *p = F* (neistinit) u sadašnjem svijetu (stanju stvari).

Tada u modalnoj logici vrijedi:

(*moguće* p) = T ako postoji bar jedan drugi svijet

◊ p =T

- neka druga situacija, neki drugi scenarij, neka druga baza znanja u kojoj je p = T.

(*nužno* p) = F jer (*nužno* p) = T samo akko je p istinit u svim svjetovima (što ovdje nije slučaj).

□p =F

U klasičnu propozicijsku i predikatnu logiku dodaju se *modalni operatori.*

Vremenska (temporalna) logike

- Prema tipu modalnosti razlikujemo logike:
  - Aletička logika potrebitost, mogućnost
  - Deontička logika: obligatornost (nužnost), dozvoljivost
  - Epistemička logika: znanje, vjerovanje
  - *Vremenska logika*: **uvijek, konačno, što_je_bilo, što_je_sad, što_će_biti**
  - . . .
- **Vremenska logika** (engl. *Temporal Logic -TL*) - višestruki pogledi:
- Propozicijska vremenska logika :
  - klasična propozicijska logika proširena vremenskim operatorima.
  - najviša razina apstrakcije u rasuđivanju.
- Vremenska predikatna logika prvoga reda (varijable, funkcije, predikati, kvantifikatori):
  - različiti tipovi vremenske logike prvoga reda
  - interpretirana-neinterpretirana (pretpostavlja ili ne strukturu),
  - globalne i lokalne varijable,
  - kvantifikacija preko vremenskih operatora ili ne.

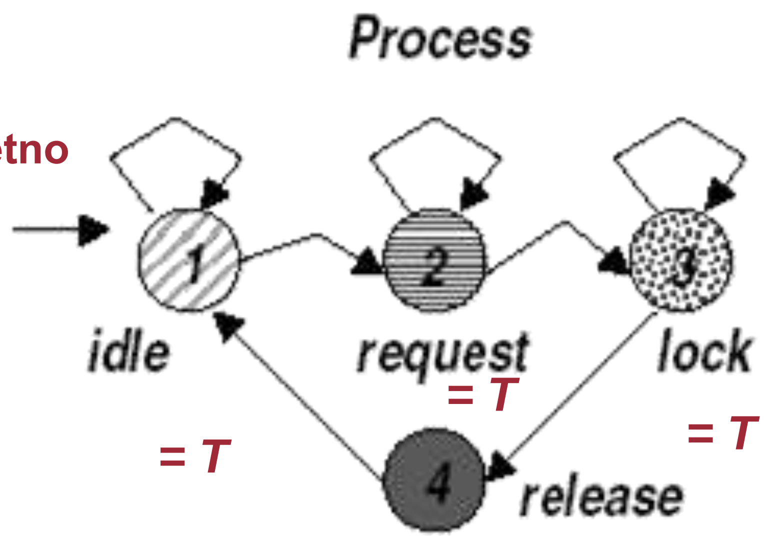
Vremenska logika: višestruki pogledi

- Globalna ili modularna:
  - Endogena i egzogena. Rasuđivanje o kompletnom sustavu ili ne.
- Vremenska logika linearnog vremena:
  - U svakom trenutku postoji samo jedan budući trenutak (jedna vremenska crta).
- Vremenska logika s grananjem vremena:
  - U svakom trenutku može postojati više različitih budućih vremenskih crta.
- Diskretno ili kontinuirano vrijeme.
  - U računarstvu uobičajeno diskretno vrijeme (sekvence stanja).
- Prošlo i buduće vrijeme.
  - Izvorno vremenska logika obuhvaća oba vremena.
  - U digitalnim sustavima uobičajeni su samo operatori budućeg vremena.
- Odabiremo:propozicijska, globalna, grananje, buduće vrijeme

Linearna vremenska struktura

- Linearna vremenska struktura opisana uređenom trojkom model implementacije (*I*) = *Kripke struktura M*
- Kripke struktura: I=M = (S, R, L) (1971. Saul Kripke)
- S : skup stanja: - skup mogućih svjetova (stanja).
- R : relacija prijelaza: - $R \subseteq S \times S$ između svjetova (stanja).
  - $\forall s \in S \ (\exists t \in S \mid (s, t) \in R)$
  - totalna binarna relacija
- L : $\mathbf{S} \rightarrow \mathbf{2}^{\mathbf{AP}}$ - funkcija označavanja stanja:
  - daje interpretaciju svih simbola iz skupa AP za stanje s (*engl. labeling*).
  - AP: skup atomičkih propozicijskih simbola
- Analogno i formalno jednako modelu automata (stroju stanja).
- U Kripke strukturi oznake na čvorovima grafa (kod automata oznake su na lukovima).

početno



$S = \{1, 2, 3, 4\}$

$R = \{(1,1), (1, 2), (2, 2), (2, 3), (3, 3), (3, 4), (4, 1)\}$

L: $L(1) = \{idle\}$

$L(2) = \{request\}$

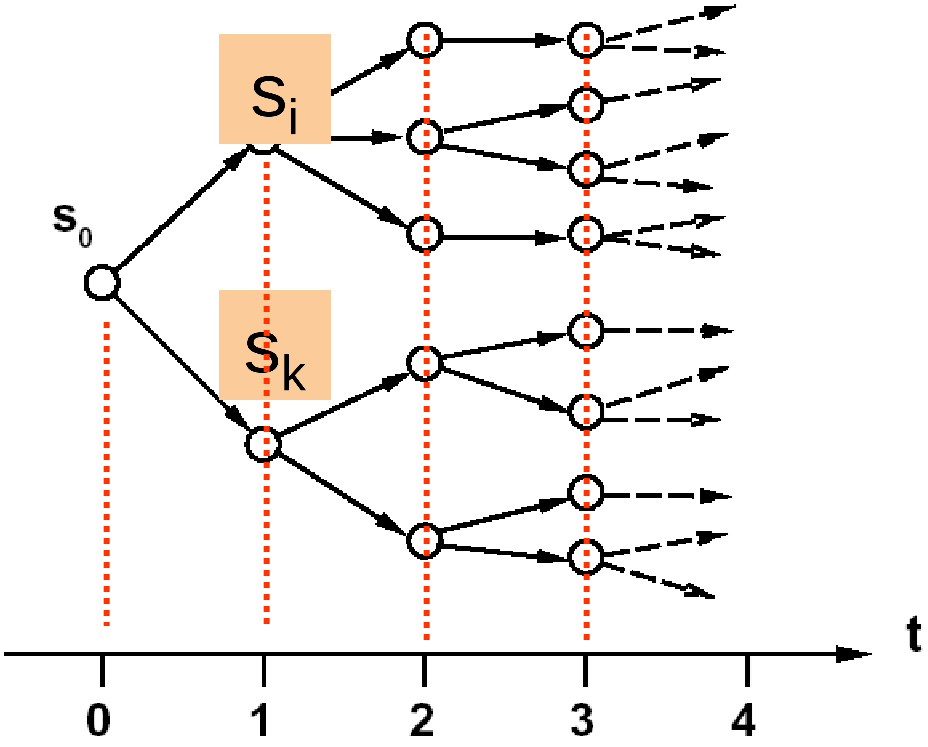
$L(3) = \{lock\}$

$L(4) = \{release\}$

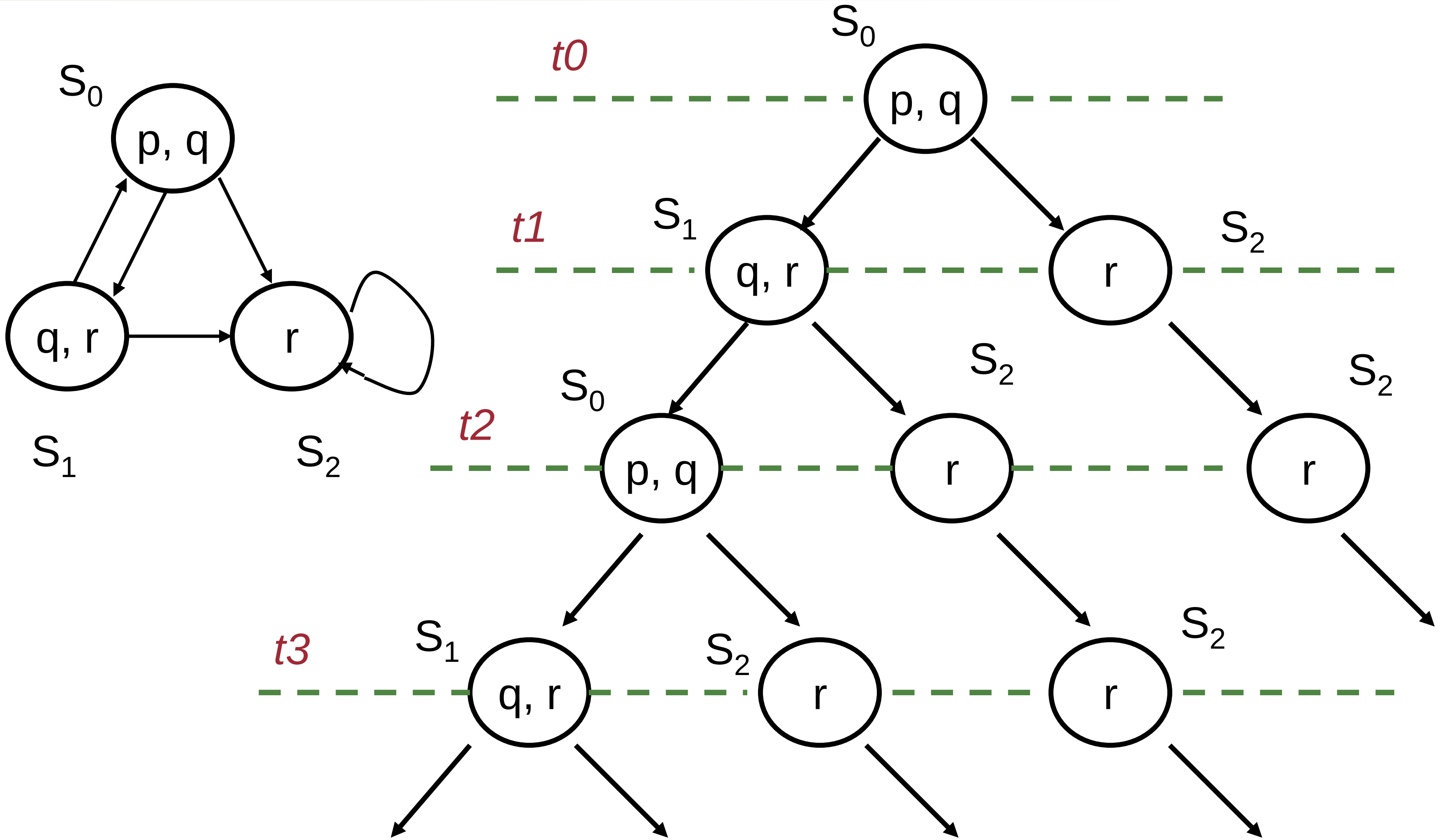
= *T* istinito u stanju

Kripke struktura M

- Tip nedeterminističkog stroja s konačnim brojem stanja
  - 1963 Saul Kripke
- Može se promatrati kao beskonačno stablo izvođenja sustava
  - “odmota” se počevši od promatranog stanja s_0
- To je *vremenska logika s grananjem*
  - *engl. Computation Tree Logic - CTL*



Primjer modela

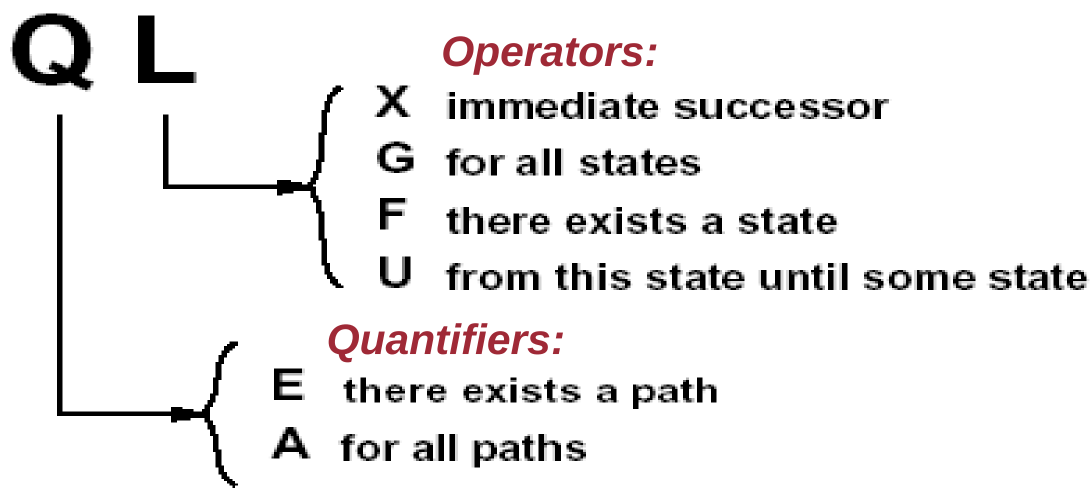


p, q, r - propozicijski simboli

S_0 - početno stanje ili stanje koje nas zanima

CTL vremenski operatori i kvantifikatori

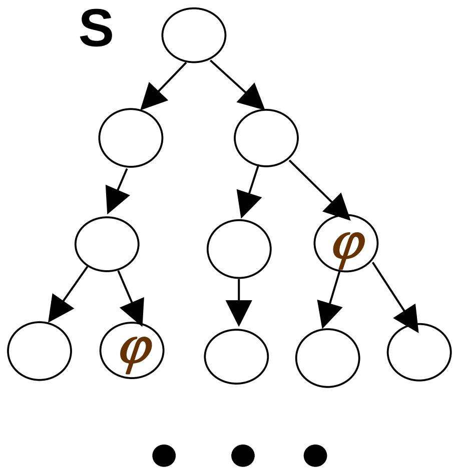
- Promatrano u kontekstu Kripke strukture
- Operatori i kvantifikatori dolaze u parovima



EX	AX
EG	AG
EF	AF
EU	AU

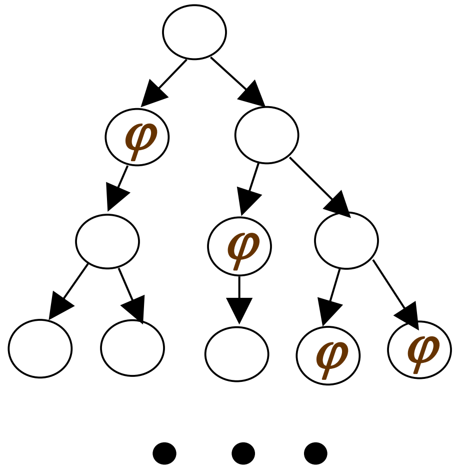
Primjeri CTL formula

- ***M – model***
- ***S – stanje***
- ***φ - formula***
- Postoji put takav da je ***φ*** eventualno istinita



EF (exists future) - ***(EF φ) = T***

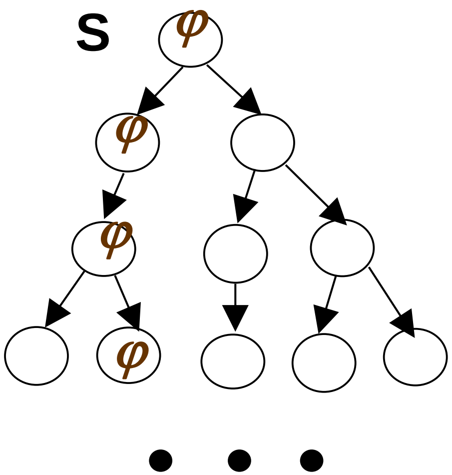
- Za sve putove vrijedi da je ***φ*** eventualno istinita



AF (all future) - ***(AF φ) = T***

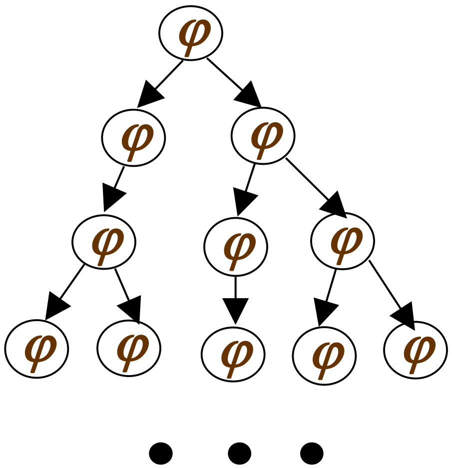
Primjeri CTL formula

- Postoji put takav da je φ istinito u svim stanjima



EG (exists globaly) - ***(EG*** φ***) = T***

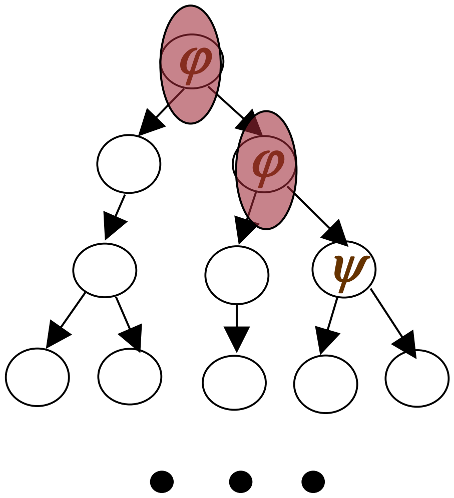
- Za sve putove vrijedi da je φ istinita



AG (allways globaly) - ***(AG*** φ***) = T***

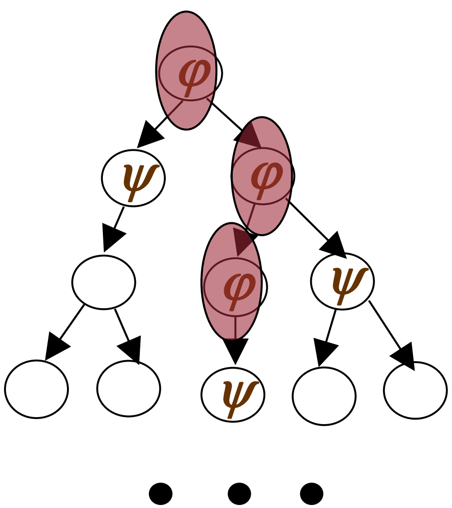
Primjeri CTL formula

- postoji put takav da je φ istinita do ispunjenja ψ



EU (exists until) - ***E*** $(\varphi\ U\ \psi) = T$

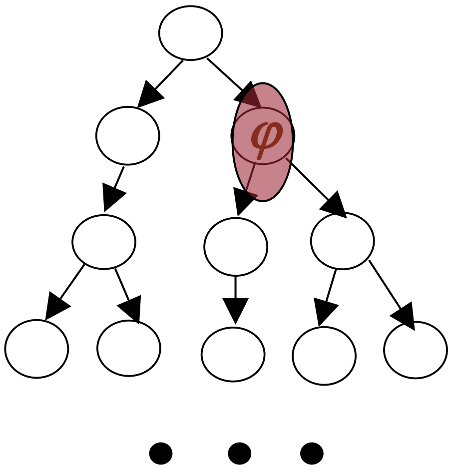
- za sve putove vrijedi da je φ istinita do ψ
  - *kakva je vrijednost* φ*?*



AU (all until) - ***A*** $(\varphi\ U\ \psi) = T$

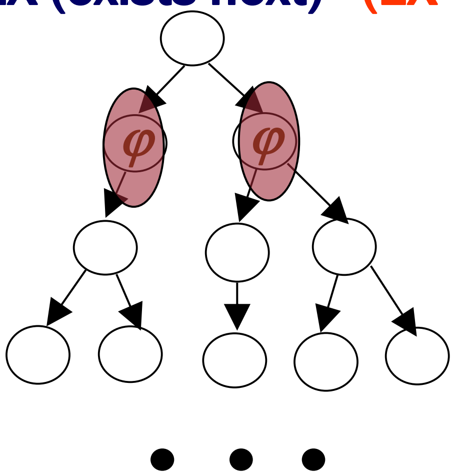
Primjeri CTL formula

- postoji put takav da je φ istinita u slijedećem stanju



EX (exists next) - (EX φ) = T

- za sve putove vrijedi da je φ istinita u slijedećem stanju



AX (all next) - (AX φ) = T

Formalna sintaksa CTL logike:

- Propozicijska formula $\in$ L = atomička formula

1. Svaka atomička formula je *formula stanja*.
2. Ako su f,g formule stanja, to su i $\neg$f, (f $\wedge$ g), (ostale se izvode)
3. Ako je f *formula puta*, E f, A f su *formule stanja*.
4. Ako su g, h *formule stanja*, tada su X g, (g U h) *formule puta*,

- Formule stanja se evaluiraju u stanjima.
- Formule puta se evaluiraju duž puta.

- Svi drugi operatori (npr. *EG*) se mogu izraziti pravilima 1 - 4.
- AU, EU - binarni operatori
- Ostali - unarni operatori
- U CTL logici formule puta ne mogu biti ugnježđene!!
  - one traže uporabu operatora E ili A da bi postale formule stanja (pravilo 3).

Primjeri CTL sintakse

- Ispravno/Dobro definirane CTL formule:

$AG\ (q \Rightarrow EG\ r)$

$EG\ p$

$E\ (p\ U\ q)$

$A\ (p\ U\ EF\ p)$

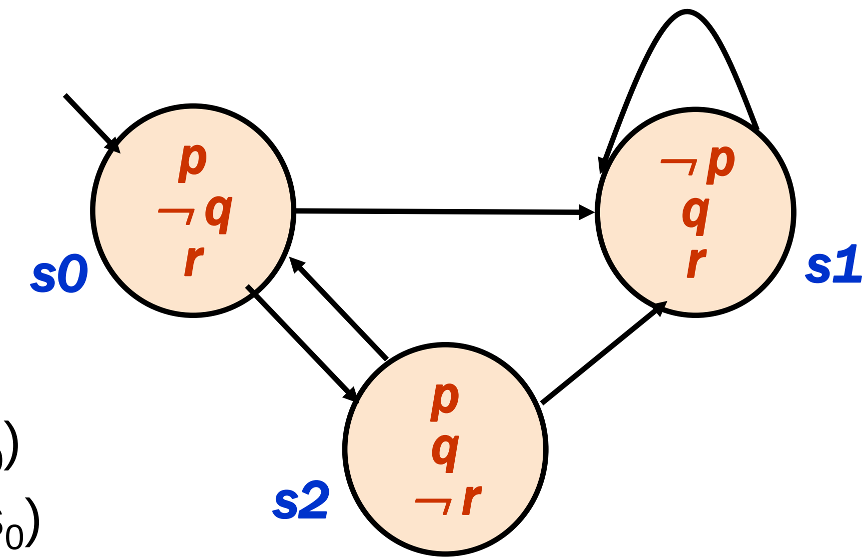
$AG\ (p \Rightarrow A\ [p\ U\ (\neg p \wedge A\ [\neg p\ U\ q]\ )\ ]\ )$

- Krivo definirane formule CTL formule:

$FG\ p$	; F i G slijede iza E ili A
$EF\ (r\ U\ q)$	; U se može upariti samo sa A ili E
	; Ex.: $EF\ E(r\ U\ q)$, $EF\ A(r\ U\ q)$
$AF\ [(r\ U\ q) \wedge (p\ U\ r)]$	; ispravan oblik je $A(\alpha\ U\ \beta)$
	; F se ne može ovdje miješati
	; $\wedge$ može biti samo unutar α ili β
	; Ex: $A\ [(p \wedge q)\ U\ (\neg r \Rightarrow q)]$

Primjer CTL

- Koja svojstva vrijede za zadani primjer?



  - (EX p)(s_0)
  - (A[p U q])(s_0)
  - (EX AF p)(s_0)
  - (A[¬ p U q])(s_0)

CTL semantika

- $M = (S, R, L)$ - model sustava (Kripke struktura)
- $M, s \models \varphi$ formula vrem. Logike φ *je istinita* u modelu M za stanje s
- $M, s \not\models \varphi$ formula vrem. logike φ *nije istinita* u modelu M za stanje s

1. $M, s \models p$ *istinita* akko $p \in L(s)$; *p je propozicijski atomički simbol*
2. $M, s \models (\varphi_1 \wedge \varphi_2)$ akko $M, s \models \varphi_1$ i $M, s \models \varphi_2$
3. $M, s \models (\varphi_1 \vee \varphi_2)$ akko $M, s \models \varphi_1$ ili $M, s \models \varphi_2$
4. $M, s \models (\varphi_1 \Rightarrow \varphi_2)$ akko $M, s \not\models \varphi_1$ ili $M, s \models \varphi$
5. $M, s \models AX\, \varphi$ akko za sve s_i takve da $s \to s_i$

   Vrijedi $M, s_i \models \varphi$ (u svakom slijedećem stanju)
6. $M, s \models EX\, \varphi$ ako za neki s_i takav da $s \to s_i$,vrijedi $M, s_i \models \varphi$

   (*u nekom slijedećem stanju*)

7. M, s $\models$ AG φ akko za sve putove $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$,

Gdje $s = s_1$ i za svaki s_i duž puta, vrijedi M, $s_i \models \varphi$

(*za sve putove koji započinju u s, obilježje φ vrijedi globalno duž puta*)

8. M, s $\models$ EG φ akko postoji put $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$,

Gdje $s = s_1$ i za svaki s_i duž puta, vrijedi M, $s_i \models \varphi$

(*postoji put koji započinje u s, takav da obilježje φ vrijedi globalno duž puta*)

9. M, s $\models$ AF φ akko za sve putove $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$,

Gdje $s = s_1$, postoji neki s_i duž puta, vrijedi M, $s_i \models \varphi$

(*za sve putove koji započinju u s, postoji neko buduće stanje u kojem vrijedi obilježje φ*)

10. M, s $\models$ EF φ akko postoji put $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$,
Gdje $s = s_1$ i za neki s_i duž puta, vrijedi M, $s_i \models \varphi$
(*postoji put koji započinje u s takav da Obilježje φ vrijedi u nekom budućem stanju)*

11. M, s $\models$ A(φ_1 U φ_2) akko za sve putove $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$,
Gdje $s = s_1$, taj put zadovoljava (φ_1 U φ_2).
φ_1 je kontinuirano istinita
Dok se ne pojavi (φ_2 = True) nekom stanju.
Formula zahtijeva da bude (φ_2 = True) U nekom budućem stanju.

12. M, s $\models$ E(φ_1 U φ_2) akko postoji put $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$,
Gdje $s = s_1$, i taj put zadovoljava (φ_1 U φ_2).
φ_1 je kontinuirano istinita
Dok se ne pojavi (φ_2 = True) u nekom stanju.
Formula zahtijeva da bude (φ_2 = True) U nekom budućem stanju.

11. i 12. : φ_1 može biti istinit ili ne u i nakon stanja u kojem φ_2=True

(semantika "until" je različita od prirodnog jezika).

φ_2 može biti istinit i prije početnog stanja s.

Za 7. do 12. : Skup budućih stanja uključuje i sadašnje stanje

(konvencija)

Posljedica:

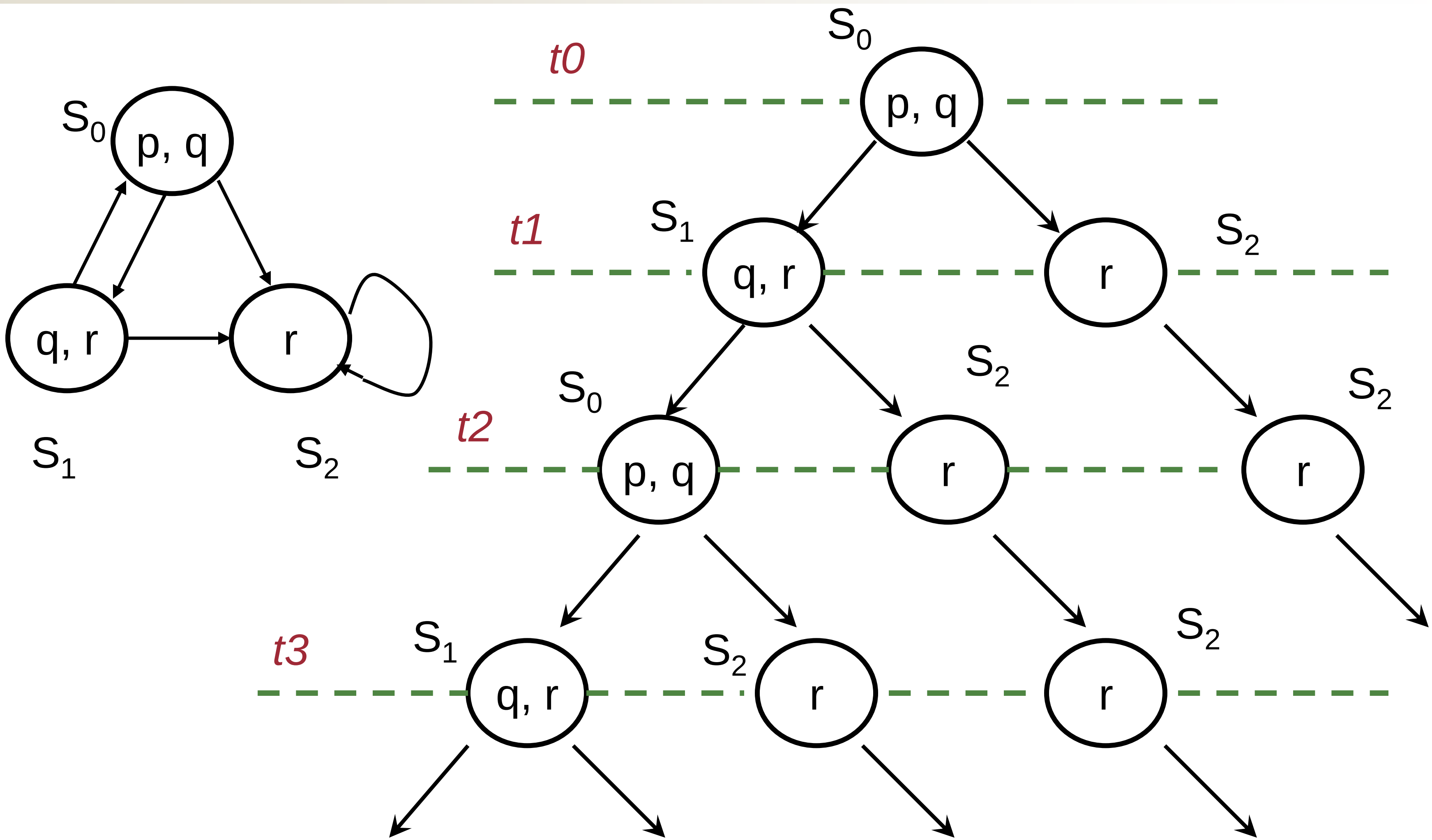
$P \Rightarrow EF\ p$ (ako p vrijedi sada, EF p također vrijedi)

$(AG\ p) \Rightarrow p$

$P \Rightarrow A(q\ U\ p)$

Ove formule su istinite u svakom stanju svakog modela.

Primjer modela



p, q, r - propozicijski simboli

S_0 - početno stanje ili stanje koje nas zanima

Istinite vremenske formule za model i stanje s_0

$M, s_0 \models (p \wedge q)$; atomi p i q su istiniti u stanju s_0

$M, s_0 \models \neg r$; atom r nije istinit u stanju s_0

$M, s_0 \models EX\,(q \wedge r)$; postoji put gdje je za slijedeće stanje vrijedi $(q \wedge r)$

$M, s_0 \models \neg AX\,(q \wedge r)$; postoji jedan put na kojem ne vrijedi za Slijedeće stanje $(q \wedge r)$

$M, s_0 \models \neg EF\,(p \wedge r)$; nema puta sa stanjem za koje vrijed $(p \wedge r)$

$M, s_0 \models AF\,r$; duž svih putova možemo dosegnuti stanje za Koje vrijedi r

$M, s_0 \models E\,[(p \wedge q)\;U\;r]$; postoji put iz s_0 na kojem u svim stanjima $(p \wedge q)$ = True, dok r=True (npr. do s_2)

$M, s_0 \models A\,[p\;U\;r]$; na svim putovima vrijedi $[p\;U\;r]$

CTL ekvivalencije

$\neg AF\ \varphi = EG\ \neg\varphi$; de Morgan

$AF\ \varphi = \neg EG\ \neg\varphi$

$EG\ \varphi = \neg AF\ \neg\varphi$

$AG\ \neg\varphi = \neg EF\ \varphi$; de Morgan

$AG\ \varphi = \neg EF\ \neg\varphi$

$\neg AX\ \varphi = EX\ \neg\varphi$; X je vlastiti dual

$AX\ \varphi = \neg EX\ \neg\varphi$

$AF\ \varphi = A\ (True\ U\ \varphi) = \neg EG\ \neg\varphi$

$EF\ \varphi = E\ (True\ U\ \varphi)$

$EG\ \varphi = \neg[\ A\ [True\ U\ \neg\varphi]]$

EG je nedjeljiv, tj. *E¬G nije ispravna CTL formula*

Notacija:

A[p U q] = *[p AU q]*

E[p U q] = *[p EU q]*

Temeljem gornjih ekvivalencija, za izračun svih CTL formula dovoljno je imati postupke za izračun EX, EG, EU = *adekvatni skup*

- *engl. adequate set*
- Postoji više adekvatnih skupova.

Primjeri: Preslikavanja prirodnog jezika u CTL

1. Moguće je doći u stanje gdje *start=T i ready=F*.

$$EF\ (start \wedge \neg ready)$$

2. Za svako stanje, ako se postavi zahtjev (za nekim resursom) biti će konačno prihvaćen (kad-tad).

$$AG\ (\ zahtjev \Rightarrow AF\ prihvaćen)$$

3. U svakom slučaju, određeni proces će *konačno* biti stalno zaustavljen

$$AF\ (AG\ zaustavljen)$$

4. Iz svakog stanja moguće je doći do stanja "restart”.

$$AG\ (EF\ restart)$$

5. Na putu prema gore, dizalo na drugom katu neće promijeniti smjer gibanja, ako postoji putnik koji želi na peti kat.

$$AG[\ (kat=2 \wedge smjer=gore \wedge pritisnuta\_tipka\_5) \Rightarrow$$
$$A\ (smjer=gore\ U\ kat=5)\ ]$$

6. Dizalo *može* ostati *stalno* stajati na trećem katu sa zatvorenim vratima.

$AG\ [\ (kat{=}3 \wedge stoji \wedge vrata{=}zatvoreno) \Rightarrow$

$EG\ (kat{=}3 \wedge stoji \wedge vrata{=}zatvoreno)\ ]$

7. Kadgod in = 1, nakon dva takta uvijek out = 1

$AG\ (in \Rightarrow AX\ AX\ out)$

8. Uvijek vrijedi: ako se pojavi signal "send" onda konačno signal "receive” postaje istinit, te do tog trenutka "send" mora ostati istinit

$AG\ (send \Rightarrow A(send\ U\ receive))$

CTL provjera modela

- *engl. CTL model checking*
- Za danu Kripke strukturu (usmjereni označeni graf)
  - I određen *skup početnih stanja S_0* ,
  - Provjeri da CTL formula zadovoljava za ta stanja:

Formalno:

$$M, S_0 \models \phi \text{ , tj.}$$

$$\forall s_0 \in S_0 \quad M, s_0 \models \phi \text{ (za svako stanje iz } S_0\text{)}$$

Postupak:

Potrebno je pronaći sva stanja koja **zadovoljavaju** CTL formulu ϕ , i **ispitati** da li je željeni podskup S_0 uključen.

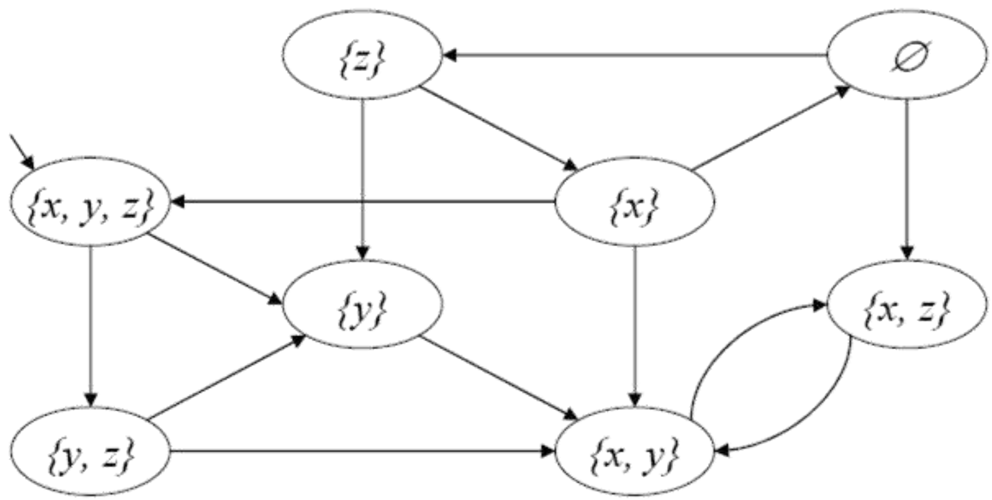
Ctl provjera modela $\Longrightarrow$ manipulacija skupovima stanja.

Primjer razrješavanja ugniježđenih operatora

- Algoritam:
  1. izračunaj stanja koje zadovoljava najugneždenija formula
     - princip iznutra prema van
  2. uporijebi rezultate za izračun drugog novoa formula
  3. ponavljaj 2
- Primjer izračuna $\mathbf{S_K(AF\ AG\ \mathit{x})}$

Example

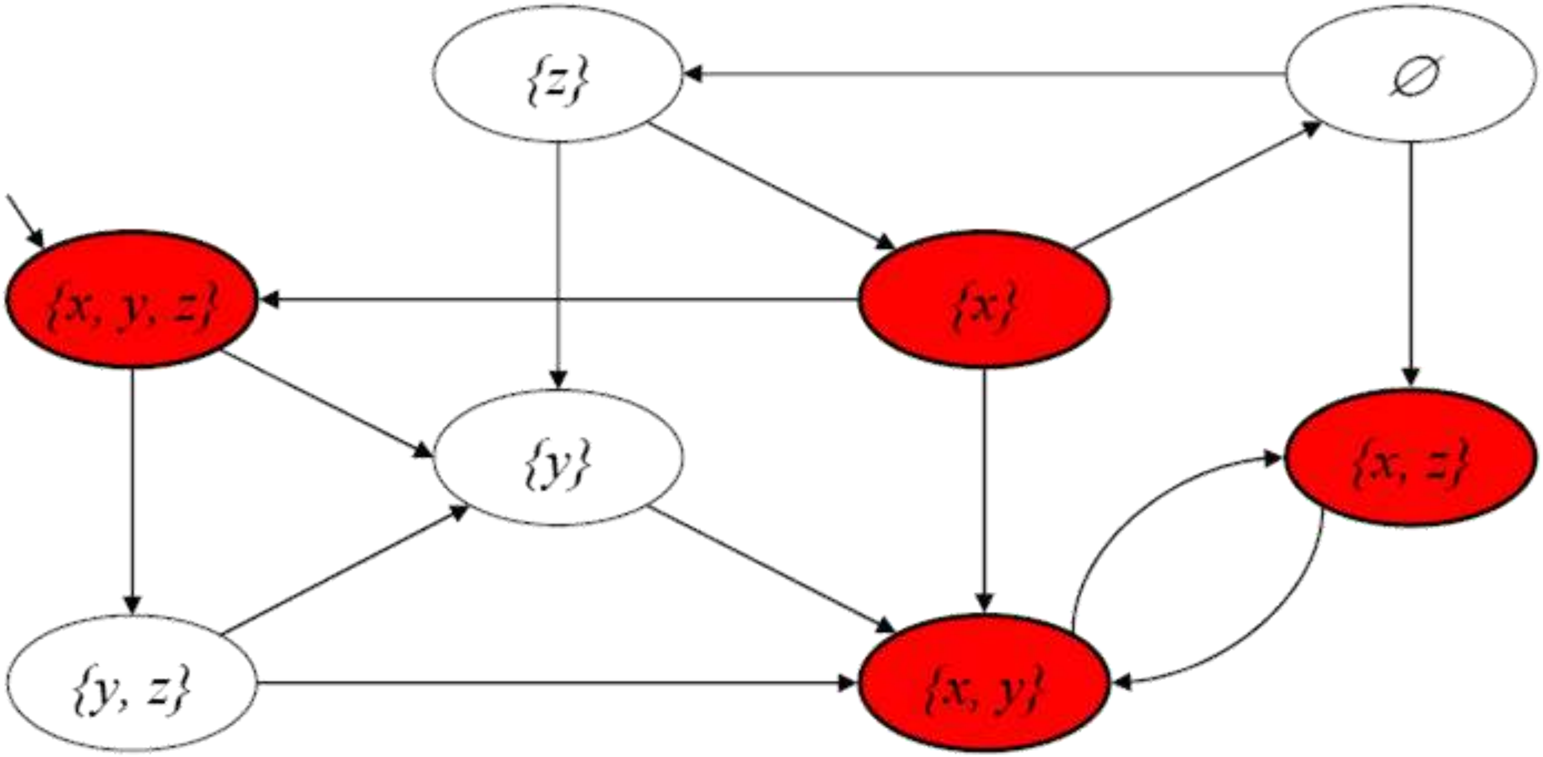
For $S_K(AF\ AG\ x)$
compute successively
- $S_K(x)$,
- $S_K(AG\ x)$, and
- $S_K(AF\ AG\ x)$



UNESCO math&dev.
TUNIS - février 2008

1. Izračunati $S_K(x)$

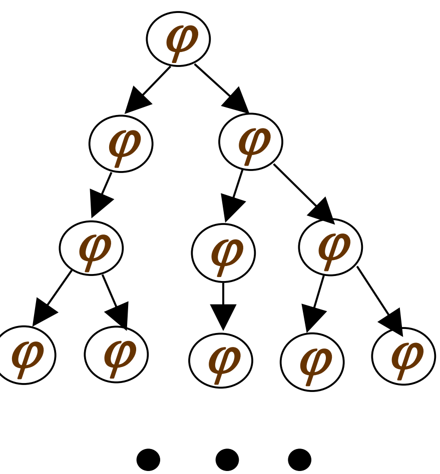
- Sva stanja u kojima postoji x

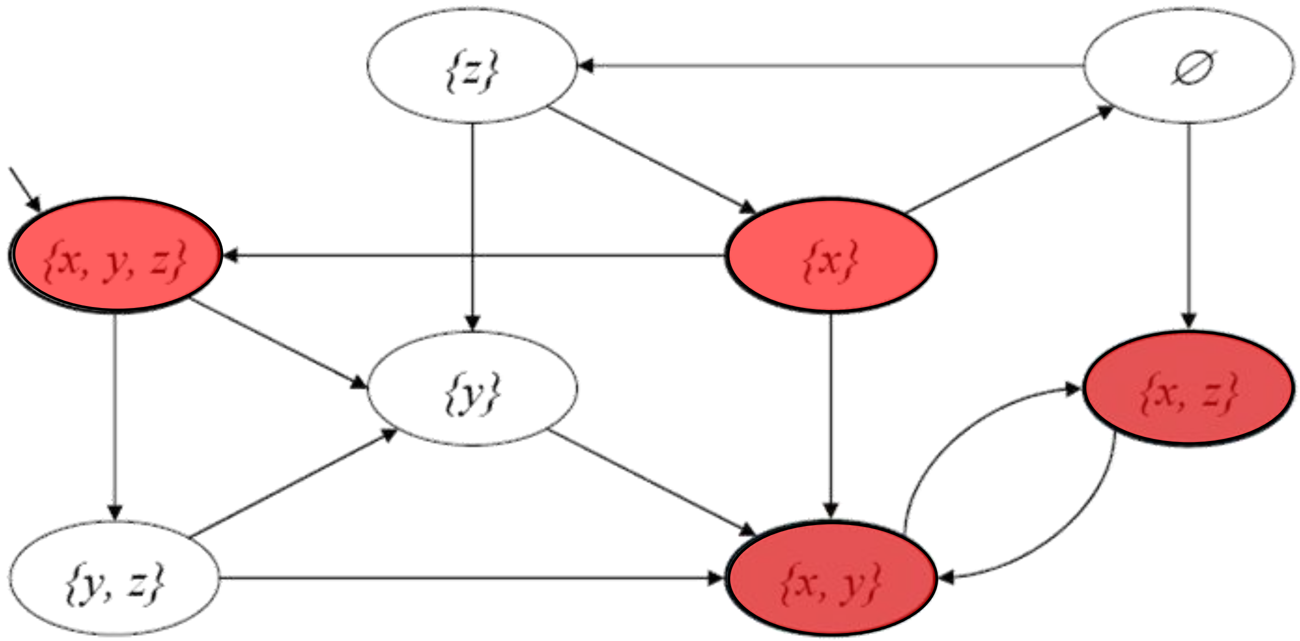


2. Izračunati $S_K(AG\ x)$

AG(x) - za sve putove vrijedi da je *x* istinita

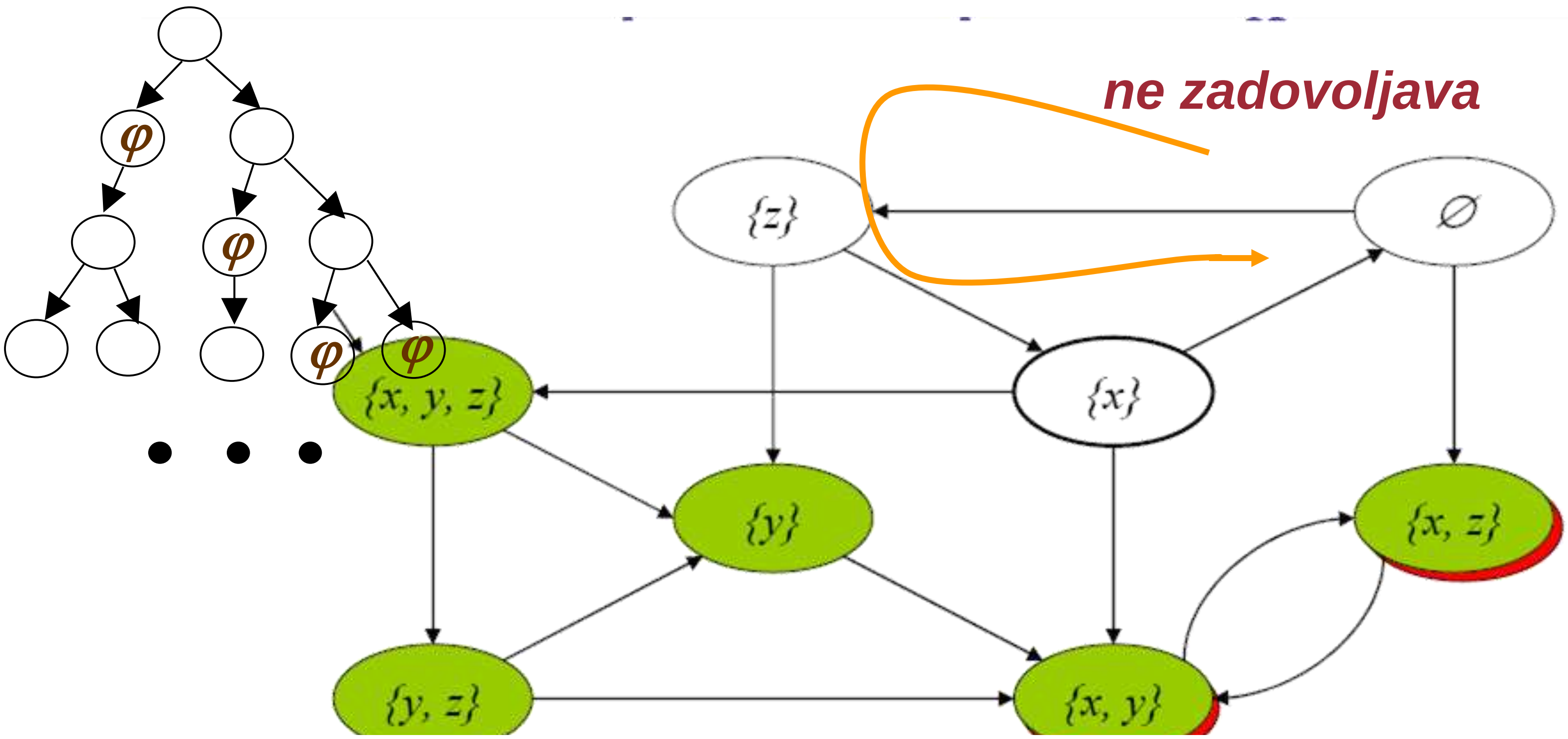
Uključuje sadašnje stanje!





UNESCO math&dev.
TUNIS - février 2008

3. Izračunati S_K(AF AG x)

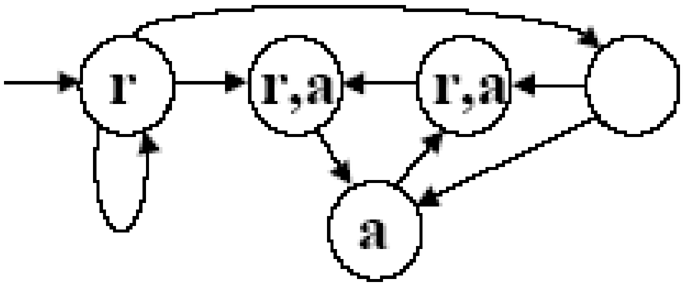


Pronaći sva stanja iz kojih se može doći do stanja AG x
NAPOMENA:
Iz tih stanja uvijek se mora moći doći do stanja AG x !

UNESCO math&dev.
TUNIS - février 2008

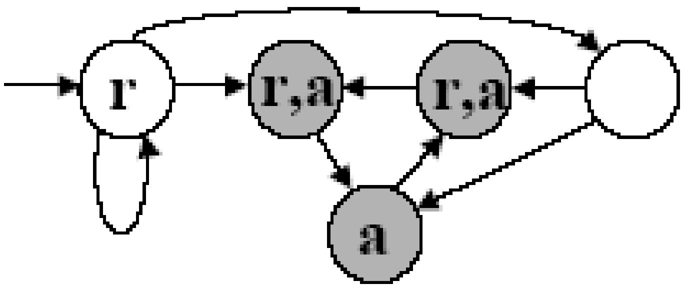
Primjer 2

Determine the set of states in machine satisfying the CTL formula: $AG(r \Rightarrow AF\ a)$ = specifikacija



= model (implementacija)

Step (1) Set of states satisfying " a "



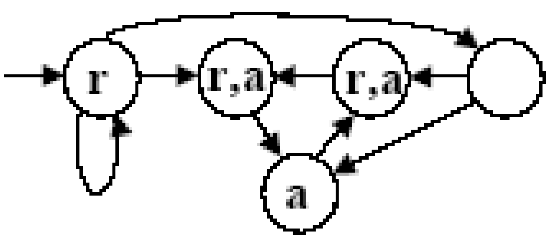
Step (2) Set of states satisfying "AF a"

Note: a state s satisfies "AF a" if either:
(i) " a " holds in s, or
(ii) "AF a" holds in every successor state of s

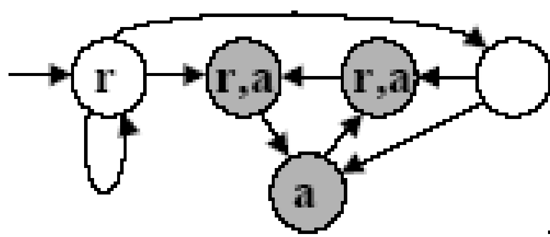
on all paths

Obtain approximations iteratively till fixpoint is reached.

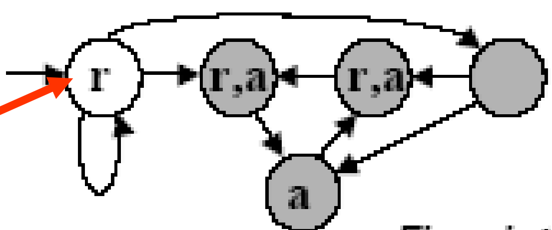
Initial approximation:



Next approximation:



Final approximation:

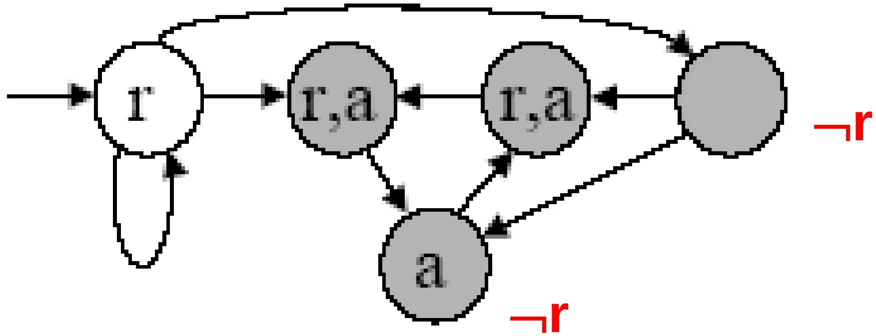


Does not hold because of this path

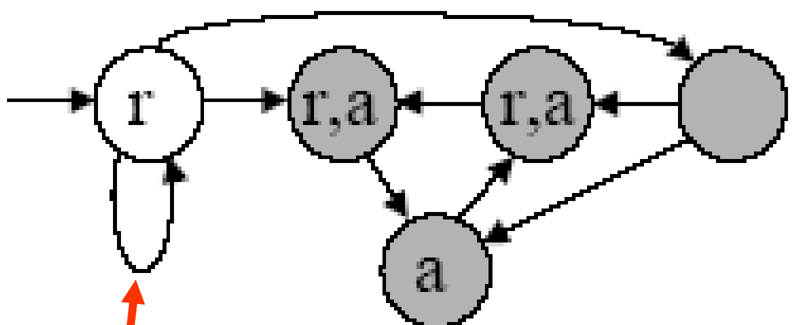
Fixpoint

Step (3) Set of states satisfying $r \Rightarrow AF\ a$

$\neg r \vee AF\ a$



Step (4) Set of states satisfying $AG(r \Rightarrow AF\ a)$

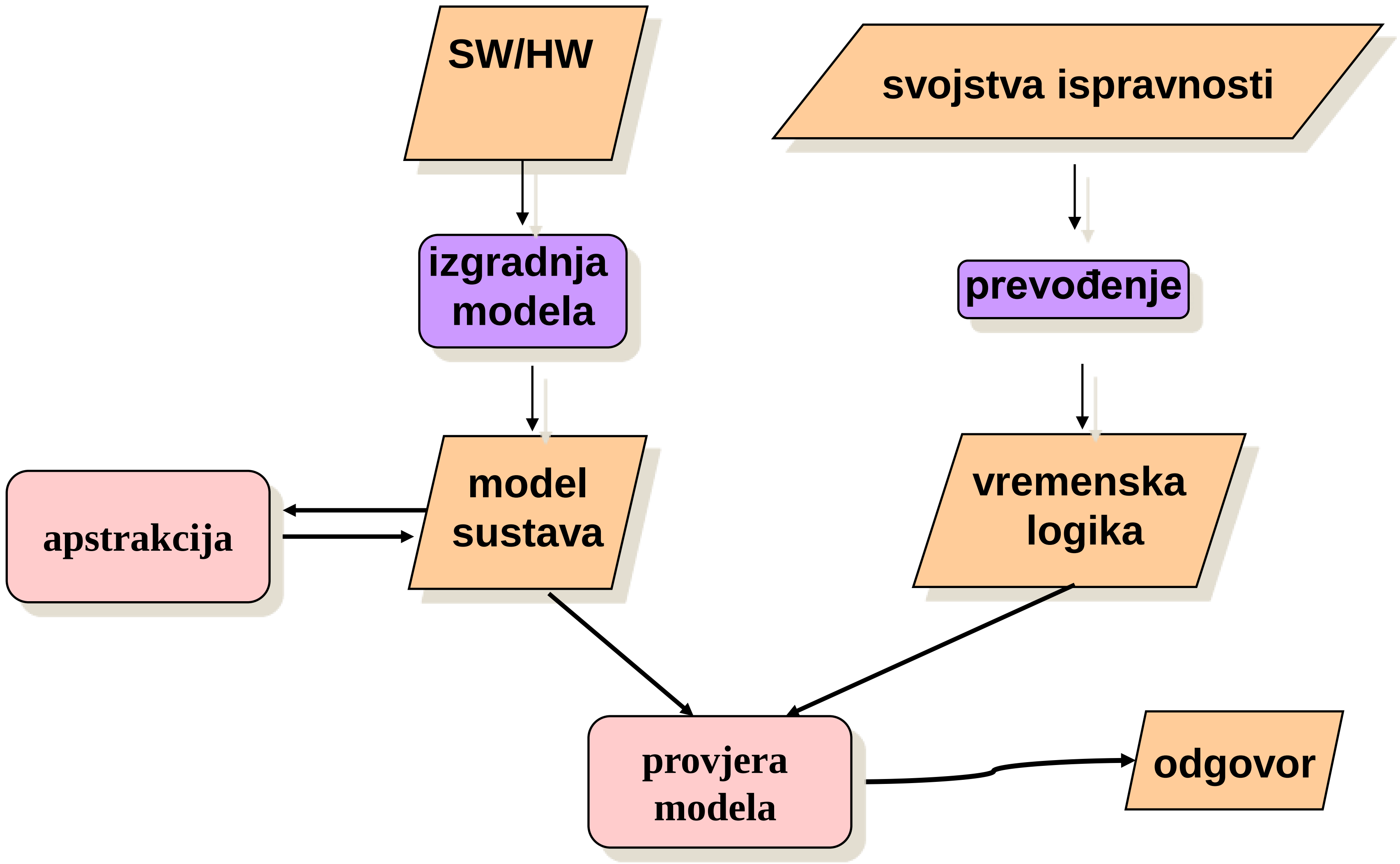


all grey states or states with all paths to grey states

Property is not true in the initial state !

Counter-example: initial state infinitely often **(not always)**

Sustavi automatske verifikacije



Formalna specifikacija

- Velik broj razvijenih formalnih specifikacija, najrasprostranjenije:
- VDM-SL (Vienna Development Method – Specification Language), IBM Research Laboratory in Vienna
  - http://www.vienna.cc/e/evdm.htm
  - Cliff B. Jones: Systematic Software Development Using VDM, by, 2nd edition, Prentice Hall,1990.
  - Fitzgerald, J.S., Larsen, P.G., Mukherjee, P., Plat, N. and Verhoef,M., Validated Designs for Object-oriented Systems. Springer Verlag 2005
- Z, PRG (Programming Research Group), University of Oxford, UK
  - http://czt.sourceforge.net/
  - Jim Woodcock, Jim Davies: Using Z: Specification, Refinement, and Proof'', Prentice Hall, 1996.
- B-Method, Jean-Raymond Abrial, France
  - http://www.methode-b.com/
  - J-R Abrial: The B-Book: Assigning Programs to Meanings, Cambridge University Press,1996
  - ZB 2000: Formal Specification and Development in Z and B, First International Conference of B and Z Users, York, UK, August 29 - September 2, 2000
  - ...

Zaključci

- Pokazana je samo jedna od mnogih *formalnih metoda*.

1. Implementacija sustava modelira se Kripke strukturom. Sustavi za verifikaciju (npr. SMV, VIS, SPIN, …) traže opis Kripke strukture u posebnim programskim jezicima.
2. Specifikacija željenog ponašanja izražava se CTL vremenskom logikom (u nekim sustavima i drugim vremenskim logikama).
3. Sustav za verifikaciju prolazi kroz sva stanja modela i provjerava da li model implementacije logički zadovoljava specifikaciju (*engl. model checking*).

- Poteškoće:
  - Precizno izraziti željeno ponašanje i modelirati strukturu.
  - Sustav za verifikaciju provjerava uvijek samo jedno željeno ponašanje.
  - Programski produkti imaju ogroman skup stanja, pa je moguća provjera samo pojedinih kritičnih dijelova.

Diskusija