Logika

FORMALNA VERIFIKACIJA PROGRAMSKE POTPORE METODOM PROVJERE MODELA (engl. *Model Checking*)

Budući da se formalnom verifikacijom nastoji dokazati logička zadovoljivost *("model implementacije zadovoljava specifikaciju")* potrebna su znanja iz slijedećih područja:

1. ***Formalna (matematička) logika*** *(posebice definicija "**logičke zadovoljivosti**" i sl.)*
2. ***Modeliranje implementacije*** *strojevima s konačnim brojem stanja (u kontekstu formalne verifikacije model je tzv. Kripke struktura) .*
3. ***Izražavanje specifikacije*** *(željenog ponašanja) vremenskom logikom kao proširenjem klasične matematičke logike.*

Formalna (matematička) logika

Različiti tipovi logike se razlikuju po sadržaju svojih "primitiva".

Dva su temeljna pogleda na logiku:

Ontološki: Što postoji u svijetu.

Epistemološki: Kakvo je stanje znanja (što agent vjeruje).

Klasična logika zasniva se na pojmu istinitosti.

Logike su formalni jezici koji predstavljaju informaciju na način da se mogu automatizirano izvoditi zaključci.

Sintaksa definira strukturu rečenice u jeziku.

Semantika definira značenje rečenica (definira istinitost rečenice u svijetu u kojem ju promatramo).

Postoji mnogo logika:

- Propozicijska i predikatna logika
- Logike višega reda
- Modalne logike
    - Epistemička logika
    - Vremenska logika
    - …
- Opisna logika

Logika određuje postupke ispravnog rasuđivanja (analogno aritmetici).

Primjer 1:

Pretpostavka (premisa) 1: 1. Svaki čovjek je smrtan.

Pretpostavka (premisa) 2: 2. Sokrat je čovjek.

Zaključak: 3. **Sokrat je smrtan.**

(Ako su istinite rečenice 1 i 2, "**logički slijedi**" rečenica 3.)

Primjer 2:

1. Svaki α ima obilježje β.

2. γ je α.

3. γ ima obilježje β.

Zaključak 3 "**Logićki slijedi**" samo na temelju oblika (forme),

a ne na temelju sadržaja (konteksta).

Matematička ili formalna logika daje sustav zaključivanja u kojem je "logički izveden" zaključak barem tako dobar kao polazne pretpostavke.

Temelj: ***formalan sustav*** (definicija formalnog sustava slijedi kasnije).

Niti jedan formalan sustav ne može osigurati istinite polazne pretpostavke

Propozicijska logika (izjava, sudova, iskaza, tvrdnji)

(engl. propositional logic, propositional calculus)

Sintaksa

Logika iskaza preslikava **deklarativne rečenice** (koje mogu biti istinite ili lažne) u sustav simbola. Npr.: "**Sokrat je mudar**." preslikava se u simbol **P**.

Sustav propozicijske logike sastoji se od:

PS: P, Q, ... PS je **prebrojiv skup atoma, simboličkih varijabli, simbola**

Logički operatori (vezice):

$\neg$	(ne, not, ~)	negacija
$\wedge$	(i, and, &)	konjunkcija
$\vee$	(ili, or, |)	disjunkcija
$\Rightarrow$	(ako, if, $\supset$, $\rightarrow$)	implikacija

$\Leftrightarrow$ (akko, iff, $\equiv$, $\leftrightarrow$) ekvivalencija

Rezervirani simboli:

F (false, $\varnothing$, 0, $\perp$) konstanta (neistinitost)

T (true, 1) konstanta (istinitost)

(), . znakovi zagrada, zareza i točke

Def. (rekurzivno) **<u>ispravno formiran (wff) složeni iskaz, ili formula</u>:**

1. Svaki atom je formula.

2. Ako su P i Q formule, onda su formule: ($\neg$P), ($\neg$Q), (P$\wedge$Q), (P$\vee$Q),

(P$\Rightarrow$Q), (P$\Leftrightarrow$Q).

<u>Semantika = pridruživanje obilježja istinitosti logičkoj formuli</u>

Pridruživanje obilježja istinitosti (T, F**)** **<u>atomičkim</u>** simbolima = **<u>Interpretacija</u>**

I: PS $\rightarrow$ BOOL

gdje je BOOL=$\{$ T, F $\}$, t.j. funkcija s kodomenom T ili F (istinito ili lažno).

Semantika **<u>dvaju složenih atomičkih simbola</u>**

prikazuje se istinitosnom tablicom.

2 suda = 2^2 = 4 interpretacije, 2^4 = 16 istinitosnih tablica

Neke važnije tablice istinitosti za povezivanje **dva** simbola:

			Implikacija	Ekvivalencija	Kontradikcija	tautologija
	P	Q	P=>Q	P$\Leftrightarrow$Q	(),$\perp$	T
I1:	T	T	T	T	F	T
I2:	T	F	F	F	F	T
I3:	F	T	T	F	F	T
I4:	F	F	T	T	F	T

Tablice istinitosti:

Truth Tables

P	Q	¬ P	P ∧ Q	P ∨ Q	P → Q	Q → P	P ↔ Q
f	f	t	f	f	t	t	t
f	t	t	f	t	t	f	f
t	f	f	f	t	f	t	f
t	t	f	t	t	t	t	t

O implikaciji (P ⇒ Q): **(1)**

To je **materijalna implikacija** i nije potpuno intuitivna prirodnom jeziku. Namjera materijalne implikacije je modelirati **uvjetnu konstrukciju**, (a ne uzročno-posljedičnu vezu), t.j.:

"ako P tada Q", t.j. ako je P istinit, tada je (P ⇒ Q) istinito samo ako je Q istinito.

Primjeri koji pokazuju **neintuitivni aspekt** materijalne implikacije:

(2 + 2 = 4) ⇒ ("Zagreb je glavni grad Hrvatske")

je istinita formula jer su prethodna (P) i posljedična (zaključna) (Q) tvrdnja istinite.

(2 + 2 = 4) ⇒ ("London je glavni grad Hrvatske")

je neistinita formula jer je posljedična tvrdnja (Q) neistinita.

Što je s formulama gdje je prethodna tvrdnja neistinita, a zaključna istinita ili neistinita:

(2 + 2 = 5) ⇒ ("Zagreb je glavni grad Hrvatske")

(2 + 2 = 5) ⇒ ("London je glavni grad Hrvatske")

U prirodnom jeziku mogli bi ovakvim formulama implikacije pridijeliti bilo istinitost ili neistinitost, a možda čak i tvrditi da ako je prethodna tvrdnja (P) neistinita, implikacija ne mora biti ni istinite ni neistinite.

U formalnoj logici prihvaćena je konvencija:

Ako je P neistinit, tada je implikacija (P $\Rightarrow$ Q) istinita, neovisno o istinitosti Q.

Zašto ima smisla implikaciju proglasiti istinitom ako je P neistinit ?

Koje su moguće opcije:

Za P = istinito suglasni smo s istinitosti implikacije (istinita ili neistinita ovisno o Q).

Za P = neistinito postoje 4 moguće tablice:

P	Q	1:	2:	3:	4:	
F	F	F	F	T	T	
F	T	F	T	F	T	što izabrati ?
T	F	F	F	F	F	
T	T	T	T	T	T	nije sporno !

Ako 1., to je konjunkcija, ako 2., to je Q, ako 3., to je ekvivalencija.

Dakle preostaje jedino 4. tablica.

Semantička pravila – izračunavanje istinitosti složene formule

Neka su: P_1, P_2 istinite, Q_1, Q_2 neistinite, **= interpretacija**

A bilo koja formula (istinita ili ne).

Istinite su formule:	**Neistinite** su formule:
$\neg Q_1$	$\neg P_1$
$(P_1 \wedge P_2)$	$(Q_1 \wedge A)$
$(P_1 \vee A)$	$(A \wedge Q_1)$
$(A \vee P_1)$	$(Q_1 \vee Q_2)$
$(A \Rightarrow P_1)$	$(P_1 \Rightarrow Q_1)$
$(Q_1 \Rightarrow A)$	$(P_1 \Leftrightarrow Q_1)$
$(P_1 \Leftrightarrow P_2)$	$(Q_1 \Leftrightarrow P_1)$
$(Q_1 \Leftrightarrow Q_2)$	$(A \wedge \neg A)$ - kontradikcija
$(A \vee \neg A)$ - tautologija	() - prazna formula

Primjer izračunavanja istinitosti složene formule s 3 propozicijska simbola

P, Q, R: $(Q \vee ((\neg Q) \wedge P) \Rightarrow R))$

Interpretacija (jedan od mogućih svjetova, ovdje 2^3 mogućih interpretacija)

Neka je jedna interpretacija I: P=T, Q=F, R=F,

izračunavanje (evaluacija) temeljem osnovnih tablica istinitosti daje formuli

neistinitu vrijednost.

Semantika uključuje interpretaciju i evaluaciju.

Pravila ekvivalencije

Definicija:

Dvije formule su semantički **ekvivalentne ili jednake** ako **imaju jednaku (istu) istinitosnu vrijednost za svaku interpretaciju.**

Ekvivalencija u slijedećim pravilima može se provjeriti tablicom istinitosti za sve interpretacije. Provjera koincidencije istinitosnih tablica **nije u općem slučaju dovoljna**, ali definicija je ispravna. Biti će objašnjeno kasnije.

$(A \wedge \neg A) = ()$	kontradikcija
$(\neg(\neg A)) = A$	dvostruka negacija
$(A \wedge A) = A$	jednaka važnost (idempotencija)
$(A \vee A) = A$	jednaka važnost
$(A \vee B) = (B \vee A)$	komutativnost
$(A \wedge B) = (B \wedge A)$	komutativnost
$((A \vee B) \vee C) = (A \vee (B \vee C))$	asocijativnost
$((A \wedge B) \wedge C) = (A \wedge (B \wedge C))$	asocijativnost
$(A \wedge (B \vee C)) = ((A \wedge B) \vee (A \wedge C))$	distributivnost
$(A \vee (B \wedge C)) = ((A \vee B) \wedge (A \vee C))$	distributivnost
$(\neg(A \vee B)) = ((\neg A) \wedge (\neg B))$	De Morganov zakon
$(\neg(A \wedge B)) = ((\neg A) \vee (\neg B))$	De Morganov zakon
$(A \Rightarrow B) = ((\neg A) \vee B)$	eliminacija uvjeta
$(A \Leftrightarrow B) = ((A \Rightarrow B) \wedge (B \Rightarrow A))$	eliminacija dvostrukog uvjeta
$(A \Rightarrow B) = ((\neg B) \Rightarrow (\neg A))$	transpozicija

Formalan logički sustav

Definiramo formalan sustav kao dvojku: $\{\Gamma, L\}$ u odabranoj logici gdje je

Γ - konačan skup ispravno definiranih (formiranih) formula (wff)

L – konačan skup pravila zaključivanja

Neka temeljna **pravila** zaključivanja (**jedan mogući skup L**)

Generiraju dodatne **istinite** formule (mehanički) bez razumijevanja konteksta (značenja).

Pogodna za strojnu primjenu.

Semantički korespondiraju sa semantikom "istinitosti".

- Ako P=T, Q=T, generiraj $(P \wedge Q) = T$ (uvođenje konjunkcije)
- Ako P=T, $(P \Rightarrow Q)$=T, generiraj Q = T ("modus ponens")
- Ako $\neg Q$=T, $(P \Rightarrow Q)$=T, generiraj $\neg P$ ("modus tolens")
- Ako $(P \wedge Q)$=T, generiraj $(Q \wedge P) = T$ (komutativnost $\wedge$)
- Ako $(P \wedge Q)$=T, generiraj P=T, Q=T ($\wedge$ eliminacija)
- Ako P=T (odnosno Q=T), generiraj $(P \vee Q) = T$ (uvođenje disjunkcije)
- Ako $[\neg(\neg P)]$=T, generiraj P=T (eliminacija negacije)

DEFINICIJE OBILJEŽJA U FORMALNOM SUSTAVU (1)

Sekvencija formula $\{\omega_1, \omega_2, \ldots, \omega_n\}$ ili pojedina formula ω_i je ***teorem* (*dokaz, dedukcija*)** iz skupa formula Γ, ako je u skupu Γ, ili se može izvesti iz Γ korištenjem pravila zaključivanja L .

$\Gamma \vdash_L \{\omega_1, \omega_2, \ldots, \omega_n\}$ sekvencija formula je teorem

$\Gamma \vdash_L \omega_i$ formula ω_i je teorem

Npr. (skup Γ sadrži dvije istinite formule): $\Gamma = \{ P, (P \Rightarrow Q) \}$

Korištenjem pravila “Modus ponens” (iz skupa dopustivih pravila L), izvodimo da je istinita nova formula Q, te je ta formula Q teorem (dokaz, dedukcija) skupa Γ.

Skup Γ je ***konzistentan*** akko (ako i samo ako)

ne sadrži formule na temelju kojih bi ω_i i $\neg\omega_i$ (istovremeno) bili teoremi.

$\Gamma = \{ P, (P \Rightarrow Q) \}$ je **konzistenatan**.

$\Gamma = \{ P, \neg P, (P \Rightarrow Q) \}$ je **nekonzistentan ili kontradiktoran** jer su P i $\neg P$ istovremeno teoremi (nalaze se u samom skupu Γ).

$\Gamma = \{ P, \neg Q, (P \Rightarrow Q) \}$ je nekonzistentan jer sadrži $\neg Q$, a pravilom "Modus ponens" može se izvesti Q, dakle $\neg Q$ i Q bi istovremeno bili teoremi.

Neka se u formalnom sustavu $\{\Gamma, L\}$ izvodi neki teorem ω_i (tražimo odgovor da li je ω_i teorem ili ne).

Sustav je ***odrediv*** **ili odlučljiv** (*engl. decidable*), akko postoji postupak, procedura ili **algoritam** koji će u konačnom vremenu odrediti ili ne teorem ω_i (dati u konačnom vremenu dati odgovor da li teorem ω_i postoji ili ne).

Formalan sustav $\{\Gamma, L\}$ **je *poluodrediv* ili poluodlučljiv** (*engl. semidecidable*), akko postoji algoritam koji će u konačnom vremenu odrediti teorem ako on postoji. Algoritam završava u konačnom vremenu s odgovorom "da" (za teorem ω_i), ali ne mora završiti u konačnom vremenu s odgovorom "ne"(t.j. ako ω_i nije teorem).

Formalan sustav je ***neodrediv*** **ili neodlučljiv** (*engl. undecidable*) ako nije odrediv ni poluodrediv.

Semantika u formalnom sustavu povezana je s:

interpretacijom (pridruživanjem istinitosti atomima) i

evaluacijom (izračunavanjem istinitosti složene formule).

Neka **interpretacija je *model*** formalnog sustava Γ ako evaluira ***sve*** njegove formule u istinito (vrijedi i za svaku formulu pojedinačno).

Npr.: interpretacija I: {P=T, Q=F, R=F} formule $(Q \vee (((\neg Q) \wedge P) \Rightarrow R))$

nije model jer ta interpretacija formuli daje neistinitu vrijednost.

Skup formula je *zadovoljiv* (*engl. satisfiable*) ako ima model (barem jedan).

Vrijedi i za pojedinačne formule.

Sukladno ranijoj definiciji**, *nezadovoljiv*** (nekonzistentan, kontradiktoran) skup formula nema nijedan model.

Skup formula Γ ***implicira*** **ili *povlači*** (*engl. entails*) formulu ω, ako je svaki model od Γ ujedno i model od ω.

Formula ω je **tada *logička posljedica*** skupa formula Γ.

$\Gamma \models \omega$ (svaki model od Γ je model formule ω)

Formula **je *valjana*** **ili *tautologija*** (*engl. valid*) ako je istinita za svaku interpretaciju i evaluaciju.

$\models \omega$ (svaka interpretacija je model formule ω)

Primjeri logičkih posljedica - 1

Svaka interpretacija koja lijevoj strani od znaka I= daje istinitost mora i desnoj strani dati istinitost.

1. $(P \wedge Q) \models P$

lijeva strana = T samo za (P=T, Q=T), samo jedan model, a to daje i desnoj strani =T, dakle gornji izraz vrijedi (P je logička posljedica ($P \wedge Q$)).

2. $(P \vee Q) \models P$

lijeva strana je istinita za (P=F,Q=T; P=T,Q=F; P=T,Q=T), ali desna za interpretaciju (P=F,Q=T) nije istinita, te P nije logička posljedica ($P \vee Q$).

3. $\{\neg Q, (P \vee Q)\} \models P$ (zarez predstavlja konjunkciju $\wedge$)

skup Γ na lijevoj strani je istinit samo za Q=F, P=T, a to daje istinitost i desnoj strani, te je P logička posljedica navedenog skupa Γ.

4. $P \models (Q \vee \neg Q)$

također vrijedi, jer za svaku interpretaciju za koju je lijeva strana istinita (P=T) i desna stana je istinita (desna strana je uvijek istinita).

Formalan sustav $\{\Gamma, L\}$ je ***ispravan*** (*engl. sound*)

ako $\Gamma \models \omega_i$ kadgod je $\Gamma \vdash_L \omega_i$,

t.j. svaka pravilima dokazana formula je ujedno i logička posljedica skupa Γ.

$\Gamma \vdash_L \omega_i$ **implicira** $\Gamma \models \omega_i$

Formalan sustav $\{\Gamma, L\}$ **je *kompletan*** (engl. *complete*)

ako $\Gamma \vdash_L \omega_i$ kadgod je $\Gamma \models \omega$,

t.j. svaku logičku posljedicu skupa Γ moguće je dokazati pravilima L.

$\Gamma \models \omega$ **implicira** $\Gamma \vdash_L \omega_i$

Većina interesantnih formalnih sustava je nekompletno, a malo ih je odredivo.

U **ispravnom i kompletnom** formalnom sustavu $\{\Gamma, L\}$ vrijedi:

$\Gamma \models \omega$ = $\Gamma \vdash_L \omega_i$ (Logička posljedica je ujedno teorem)

"Zaključak" inje jednoznačan termin i treba ga izbjegavati.

Umjesto "zaključak" rabi se **teorem i/ili logička posljedica**.

Propozicijska logika je ispravna, kompletna i odrediva (npr. preslikavanjem u tablicu istinitosti), jer operira **s konačnim** skupom simbola.

Primjeri:

Prioritet logičkih operatora:

Najviši:	$\neg$	negacija
	$\wedge$	konjunkcija
	$\vee$	disjunkcija
	=>	implikacija
Najniži:	$\Leftrightarrow$	ekvivalencija

Formule:	Obilježja:
1. P	**zadovoljiva** ali ne i valjana (interpretacija P=T je model, dok interpretacija P=F nije model).
2. $(P \vee \neg P)$	**tautologija** (valjana), sve interpretacije (dvije) P=T, P=F, su modeli (formula je istinita).
3. $(P \wedge \neg P)$	**kontradiktorna** (nezadovoljiva), nema modela.
4. ()	**kontradiktorna** (nezadovoljiva).
5. $P \Rightarrow (Q \Rightarrow P)$	**tautologija** (valjana), sve interpretacije (ima ih 4: FF, FT, TF, TT) su modeli.
6. $(P \wedge Q)$	**zadovoljiva**. Ima samo jedan model: P=T, Q=T.

Dodatno čvršće objašnjenje semantičke ekvivalencije:

Ranija definicija: Dvije formule su **semantički ekvivalentne ili jednake** ako imaju **jednaku (istu) istinitosnu vrijednost za svaku interpretaciju I.**

Npr. Da li su ekvivalentne dvije formule: $((P \wedge Q) \Rightarrow P)$ i $(R \vee \neg R)$

jesu sukladno gornjoj definiciji (obje su valjane -- tautologije), ali usporedba istinitosnih tablica nema smisla (simboli i tablice su različite).

Definicija ekvivalencije preko pojma logičke posljedice ($\models$)

Ako dvije ekvivalentne formule imaju jednaku istinitosnu vrijednost za svaku interpretaciju, može se definirati:

Dvije formule α i β su semantički ekvivalentne (oznake $(\alpha \Leftrightarrow \beta)$ ili $(\alpha \equiv \beta)$) akko vrijede (istinite su) logičke posljedice: $(\alpha \models \beta)$ i $(\beta \models \alpha)$.

Ranija tablica pravila ekvivalencije daje: $(\alpha \Leftrightarrow \beta) = (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

Ako su α i β ekvivalentne, formula $((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ mora biti uvijek istinita:

$\models ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$

Semantička ekvivalencija je na taj način identična **dokazljivoj** ekvivalenciji:

Ako želiš dokazati ekvivalentnost, dokaži da je $((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ tautologija, odnosno da je njena negacija nezadovoljiva.

TEOREM DEDUKCIJE

Dokazivanje logičke posljedice preko (ne)zadovoljivosti

Formula ψ je **logička posljedica** formule φ, t.j. $\varphi \models \psi$, akko je formula $(\varphi \Rightarrow \psi)$ tautologija (valjana).

Dokaz:

Akko je $(\varphi \Rightarrow \psi)$ tautologija (uvijek istinita), onda iz tablice za implikaciju proizlazi da kada je φ istini i ψ mora biti istinit. To je upravo definicija logičke posljedice.

φ	ψ	$(\varphi \Rightarrow \psi)$
F	F	T
F	T	T
T	F	F
T	**T**	**T**

Budući da $(\varphi \Rightarrow \psi)$ mora bit tautologija, to njena negacija

$\neg(\varphi \Rightarrow \psi) = \neg(\neg\varphi \vee \psi) = (\varphi \wedge \neg\psi)$ mora biti nezadovoljiva. Dakle:

$\varphi \models \psi$ akko je $(\varphi \wedge \neg\psi)$ nezadovoljiva

Normalni oblici logičkih formula

Svaka propozicijska formula može se preslikati (ekvivalentna je) formuli u **disjunkcijskom normalnom obliku (DNF) :**

$(k1_1 \wedge \ldots \wedge k1_n) \vee (k2_1 \wedge \ldots \wedge k2_m) \vee \ldots \vee (kp_1 \wedge \ldots \wedge kp_r)$

Svaka propozicijska formula može se preslikati (ekvivalentna je) formuli u **konjunkcijskom normalnom obliku (CNF) :**

$(k1_1 \vee \ldots \vee k1_n) \wedge (k2_1 \vee \ldots \vee k2_m) \wedge \ldots \wedge (kp_1 \vee \ldots \vee kp_r)$

gdje su:

k_i **= literal** (negirani ili nenegirani atomički simbol - atom)

klauzula = disjunkcija literala. Npr.: $(k2_1 \vee \ldots \vee k2_m)$

CNF = konjunkcija klauzula

Zašto normalni oblici ?

Daju brze odgovori na neka česta pitanja.

DNF nam govori **da li je formula zadovoljiva**. Ako su sve disjunkcije neistinite ($\perp$), ili sve sadrže komplementarne literale (npr. ($A \wedge \neg A$)) ne postoji niti jedan model za taj DNF. Inače je formula zadovoljiva.

CNF nam govori **da li je ili nije formula tautologija** (valjana, uvijek istinita). Ako sve klauzule sadrže istinitost (T) ili sve sadrže komplementarne literale (npr ($A \vee \neg A$)) formula je tautologija. Inače, za formulu postoji barem jedna interpretacija (pridruživanje istinitosti atomičkim simbolima) koja nije model (ne zadovoljava formulu) pa formula nije tautologija.

Preslikavanje **CNF u DNF** i obrnuto je računalno vrlo skupo (vremenski i prostorno).

<u>SAT problem (zadovoljivost) - temeljni NP problem - 1</u>

Tražimo model skupa formula Γ (interpretaciju koja evaluira sve formule u skupu Γ u istinito. To je ekvivalentno traženju modela **jedne** složene formule koja se sastoji iz **konjunkcije svih formula u Γ.**

Γ skup formula je najčešće dan u **CNF** obliku:

$(k1_1 \vee \ldots \vee k1_p) \wedge (k2_1 \vee \ldots \vee k2_r) \wedge \ldots \wedge (kp_1 \vee \ldots \vee kp_s)$

Iscrpna procedura rješavanja **CNF** SAT problema sistematski pridjeljuje istinitosne vrijednosti atomičkim propozicijskim simbolima.

Za **n atoma 2^n pridruživanja.**

Eksponencijalna složenost, računalno neizvedivo u općem slučaju.

Za **DNF – polinomska složenost** jer postoji konačan broj literala, a dovoljno je pronaći zadovoljivost u samo jednom disjunkcijskom članu.

CNF 2SAT - polinomska kompleksnost (do 2 literala u klauzuli)

CNF 3SAT **- NP kompletno** (3 literala u klauzuli)

Zadovoljivost formule u CNF obliku s 3 i više literala je NP kompletno.

Mnogi stohastički algoritmi troše eksponencijalno vrijeme u najgorem slučaju, ali polinomsko u srednjem (očekivanom).

<u>Teorem dedukcije – dokazivanje SAT rješavačem</u>

Neka **istinite** formule predstavljaju skup Γ :

1. **P**
2. **$(P \Rightarrow Q)$**
3. **$(Q \Rightarrow S)$**

U CNF obliku: **$\Gamma = [\,(P) \wedge (\neg P \vee Q) \wedge (\neg Q \vee S)\,]$**

Da li je **S** logička posljedica skupa Γ : $\Gamma \models \mathbf{S}$?

Teorem dedukcije:

S je logička posljedica Γ ako je $(\Gamma \wedge \neg \mathbf{S})$ nezadovoljiva.

Skupu Γ dodajemo negaciju formule koju želimo dokazati **(¬S**):

[(P) ∧ (¬P ∨ Q) ∧ (¬Q ∨ S) ∧ (¬S)]

SAT sustavom pokušamo naći bar jedan model (zadovoljivost).

Ako SAT sustav pokaže da formulu nije moguće zadovoljiti (nema modela), zaključujemo:

S je doista logička posljedica skupa Γ.

<u>Konverzija propozicijske formule u CNF oblik</u>

Svaka formula u propozicijskoj logici može se konvertirati u

konjunkciju klauzula **(CNF**):

Npr: $\neg(P \Rightarrow Q) \vee (R \Rightarrow P)$

1. Eliminiraj implikaciju uporabom ekvivalentnog "$\vee$" oblika:

$\neg(\neg P \vee Q) \vee (\neg R \vee P)$

2. Reduciraj doseg negacije (pomak u desno) uporabom

DeMrorgan pravila, te eliminiraj dvostruke negacije:

$(P \wedge \neg Q) \vee (\neg R \vee P)$

3. Pretvori u CNF asocijativnim i distribucijskim pravilima:

$(P \vee \neg R \vee P) \wedge (\neg Q \vee \neg R \vee P)$,

te dalje:

$(P \vee \neg R) \wedge (\neg Q \vee \neg R \vee P)$ **= CNF oblik**

<u>PREDIKATNA LOGIKA</u> (Logika predikata prvoga reda - FOPL)

(engl. predicate logic, predicate calculus, first order predicate logic)

G.Frege, A.Tarski

1. **P**: Svi ljudi su smrtni.

2. **Q**: Sokrat je čovjek.

3**. R**: Sokrat je smrtan.

U propozicijskoj logici nikako se iz 1 i 2 ne može zaključiti 3.

FOPL uvodi objekte, relacije, obilježja, funkcije (pobliži opis izjave).

Sintaksa:

Atomički predikat:

(pred_simb t1 t2 … tn) – infiks notacija (LISP)

pred_simb(t1 t2 … tn) - prefiks notacija (Prolog)

pred_simbol: osnovno obilježje u rečenici (predikat)

t_i **= članovi**: objekti ili odnosi u rečenici

Članovi (t_i) :

Konstante: objekti u nekom svijetu (blok1, sokrat, …).

Rezervirane konstante: T, F.

Varijable: razred objekata ili obilježja;

mogu poprimiti vrijednosti iz svoje domene**;** (Npr.: X, Y, …).

Funkcije: veza između objekata - (fun_simb t1 t2 … tn)

Npr.: (cos X), (brat_od abel kain)

Formalna def. člana:

1. Konstanta je član.

2. Varijabla je član.

3. Ako je **fun_simb** funkcijski simbol sa n-argumenata, a t_1, t_2, …, t_n su članovi,

tada je **(fun_simb t_1 t_2 … t_n)** član.

Logički operatori (vezice): $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$

Kvantifikacijski simboli (uz varijable, pobliže određuju istinitost rečenice):

$\exists$ (postoji, za_neki, exist) - egzistencijski ili partikularni kvantifikator (barem jedan).

$\forall$ (za_svaki, svi, for_all) - univerzalni kvantifikator (svi), ima središnju ulogu u izražavanju generalizacije**.**

Ispravno definiran složeni predikat ili formula (wff):

1. svaki atomički predikat je formula.

2. ako je S_i formula, tada su formule:

$(\neg S)$, $(S_1 \wedge S_2)$, $(S_1 \vee S_2)$, $(S_1 \Rightarrow S_2)$, $(S_1 \Leftrightarrow S_2)$.

3. ako je X varijabla, a S formula, tada su formule: $\exists X\ S(X)$, $\forall X\ S(X)$.

(oznaka S(X) = formula S u kojoj postoji varijabla X)

Negirani ili nenegirani atomički predikat naziva **se literal**.

Dopuna pravilima ekvivalencije (P(X), Q(X) su wff s varijablom X):

$(\neg(\forall X\ P(X))) = \exists X\ (\neg P(X))$ - analogno De Morgan

$(\neg(\exists X\ Q(X))) = \forall X\ (\neg Q(X))$

$\exists X\ P(X) = \exists Y\ P(Y)$ - simbol varijable nije bitan, ali je bitan

$\forall X\ P(X) = \forall Y\ P(Y)$ doseg, uvijek unutar jedne formule

Primjer ispravno definirane složene formule u infiks notaciji:

$(\forall X\ \forall Y\ (((\text{otac } X\ Y) \vee (\text{majka } X\ Y)) \Rightarrow (\text{roditelj } X\ Y)))$

Semantika predikatne logike (1)

Skup wff odnosi se na neku domenu razmatranja D.

- **Interpretacija** *I* je proces preslikavanja elemenata iz domene D svakoj pojedinoj konstanti, varijabli, i funkciji, te atomičkom predikatu, tako da:
  - Simbolu T uvijek je pridružena istinita vrijednost.
  - Simbolu F uvijek je pridružena neistinita vrijednost.
  - Svakoj konstanti pridruži se jedan element iz D.
  - Svakom funkcijskom simbolu pridruži se jedan element iz D.
  - Svakoj varijabli se pridruži neprazan podskup iz D (dozvoljene supstitucije).
- Svaka funkcija f, sa m argumenata, definira interpretacijom **i evaluacijom** preslikavanje iz D^m u D, t.j.: f: $D^m \rightarrow D$ (pridruživanje jednog elementa iz D).
- Svaki predikat P, s brojem članova n, definira **interpretacijom i evaluacijom** svojih članova preslikavanje iz D^n u $\{T, F\}$, t.j. P: $D^n \rightarrow \{T, F\}$ (istinito ili ne).
- Vrijednosti wff formula složenih logičkim operatorima date su odgovarajućim istinitosnim tablicama.

- Vrijednost ∀X P(X) je T, ako P(X) je T, za sve vrijednosti X dane sa *I*,

  a F inače.

- Vrijednost ∃X P(X) je T, ako P(X) je T, barem za jednu vrijednost X danoj sa *I*, a F inače.

Određivanje istinitosti wff svodi se na **interpretaciju + evaluaciju**

Primjeri pridruživanja istinitosti:

1. **(prijatelj ivan ana)**

   predikat je T, ako u D postoji objekt Ana koja je prijatelj Ivanu.

2. Neka je X je domena prirodnih brojeva

   ∀X (veci X 10) atomički predikat je F

   ∃X (veci X 10) atomički prediakt je T

∀ - u određivanju T potrebne sve supstitucije varijable

(problem ako je domena beskonačna)

∃ - u određivanju T potrebna jedna supstitucija za koju T

(problem ako je domena beskonačna i predikat F)

Skup svih istinitih predikata iz domene D = **<u>stanje svijeta</u>** (*engl. state of the world*).

<u>Primjeri preslikavanja prirodnog jezika u formule predikatne logike</u>

1. "Nitko nije savršen." (infiks notacija)

   ¬∃X (savrsen X), ili ∀X (¬(savrsen X)), t.j. "svi su nesavršeni"

2. "Svi košarkaši su visoki." (infiks notacija

   ∀X ((kosarkas X) ⇒ (visok X))

"za svaki X u domeni razmatranja vrijedi da ako je (X) košarkaš tada je visok"

<u>Ispravna uporaba univerzalnog kvantifikatora ∀ (1)</u>

Neka je okvir razmatranja (skup objekata): **{ Garfield, Feliks, računalo**}

Preslikaj u pred. logiku: "**Sve mačke su sisavci**."

Za <u>sve</u> objekte u okviru razmatranja vrijedi: ako su mačke tada su sisavci.

∀ x [mačka(x) ⇒ sisavac(x)] (prefiks notacija)

Ispravna uporaba univerzalnog kvantifikatora ∀ (2)

∀ x [mačka(x) ⇒ sisavac(x)] **(vrijedi za sve** objekte x)

Dokaz: Supstitucija svih objekata u formulu (**konjunkcija formula jer ∀):**

[mačka(Garfield) ⇒ sisavac(Garfield)] ∧ [mačka(Feliks) ⇒ sisavac(Feliks)] ∧ [mačka(računalo) ⇒ sisavac(računalo)] ∧ ... (ostali objekti ako ih ima)

prva [ ]: T (vidi tablicu za ⇒)

druga [ ]: T (vidi tablicu za ⇒)

treća [F ⇒ T] = T (vidi tablicu za ⇒) **pa je i treća formula = T !!!!!**

Ako bi preslikali: **∀ x [mačka(x) ∧ sisavac(x)]**

Supstitucija svih objekata daje:

[mačka(Garfield) ∧ sisavac(Garfield)] ∧ [mačka(Feliks) ∧ sisavac(Feliks)] ∧

[mačka(računalo) ∧ sisavac(računalo)] ∧ … (ostali objekti ako ih ima)

mačka(računalo) = F **- daje neistinitu cijelu formulu !!**

Ispravna uporaba egzistencijskog kvantifikatora ∃ (1)

Neka je okvir razmatranja (kao prije): { Garfield, Feliks, računalo}

Preslikaj u predikatnu logiku: "**Garfield ima brata koji je mačka**."

Postoji **barem jedan** (neki) objekt i **takav da su mu obilježja istinita**.

∃x [brat(x, Garfield) ∧ mačka(x)]

Dokaz supstitucijom svih objekata u formulu (**disjunkcija formula jer ∃**):

[brat(Garfield, Garfield) ∧ mačka(Garfield)] ∨

[brat(Feliks, Garfield) ∧ mačka(Feliks)] ∨

[brat(računalo, Garfield) ∧ mačka(računalo)] ∨ … (ostali ako ih ima)

Prva [ ] neistinita, ali idemo dalje jer su […] povezane **disjunkcijom**.

Drugi red istinit, cijela formula je istinita (dalje ne moramo ispitivati).

Ispravna uporaba egzistencijskog kavantifikatora ∃ (2)

Ako bi preslikali: **∃x [brat(x, Garfield) ⇒ mačka(x)]**

Supstitucija svih objekata u disjunkciju formula daje:

[brat(Garfield, Garfield) ⇒ mačka(Garfield)] ∨

[brat(Feliks, Garfield) ⇒ mačka(Feliks)] ∨

[brat(računalo, Garfield) $\Rightarrow$ mačka(računalo)] $\vee$ … (ostali objekti ako ih ima)

Implikacija je istinta ako je atomički izraz na lijevoj strani neistinit !

Npr. ako je**: [brat(računalo, Garfield) $\Rightarrow$ mačka(računalo)]** istinito,

cijela je formula istinita !!

Egzistencijski kvantificirana implikacijska formula je istinita ako

u okviru razmatranja postoji barem jedan objekt

za koji je premisa implikacije **neistinita** (desna strana može biti T ili F).

Takva rečenica **ne daje nikakvu potvrdnu informaciju**.

Zaključak: $\forall$ **ide uz** $\Rightarrow$

$\exists$ **ide uz** $\wedge$

Ispitni zadatak iz predikatne logike

Preslikaj u predikatnu logiku: "Niti jedan student ne sluša sve predmete."

Rješenje: Definiramo predikate (formalna logika ne definira predikate):

S(x) - x je student (prefiks notacija)

L(x) - x je predmet

B(x, y) - x sluša y

Prioriteti logičkih operatora: $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$.

Preporuka: koristi zagrade za prioritet i doseg kvantif.

a) "Ne postoji x koji je student i takav da sluša sve predmete."

$\neg\exists x [S(x) \wedge \forall y (L(y) \Rightarrow B(x, y))]$

b) "Za svaki x vrijedi: ako je student postoji predmet (bar jedan) koji ne sluša"

$\forall x [S(x) \Rightarrow (\exists y (L(y) \wedge \neg B(x, y)))]$

Evidentno: 1) $\exists$ **ide uz** $\wedge$, $\forall$ **ide uz** $\Rightarrow$

2) pomicanjem negacije u a) slijedi b) - DeMorgan

Svojstva predikatne logike

Zadovoljivost

Model

Logička posljedica

Kontradiktornost

Pravila zaključivanja

Logika predikata višega reda

Predikatna logika:

> Kvantifikacija samo varijabli (objekata u domeni D), a ne na odnose (predikatni ili funkcijski simbol) u domeni D.

Logika višega reda:

> ∀(Voli) (Voli ivo ana)

kvantifikacija na predikatnom (ili funkcijskom) simbolu.

Obilježja nekih logika:

- Većina interesantnih formalnih logičkih sustava je nekompletna, a vrlo malo ih je odredivo.
- **Propozicijska logika je:**
  - **Ispravna, kompletna i odrediva** (npr. preslikavanjem u tablicu istinitosti), jer operira s konačnim skupom simbola.
- **Predikatna logika je:**
  - **Poluodrediva** (ako teorem postoji, dokazat će se, a ako ne postoji može se ali i ne mora dokazati).
  - "Čista" (npr. bez aritmetike) predikatna logika je **ispravna i kompletna** (Gödel).
- **Tipične primjene formalne logike**
- 1. Matematika (dokazivanje teorema)
- 2. Formalna logika (dopuna teorije)
- 3. Zagonetke (imitacija razumskog rasuđivanja)
- **4. Oblikovanje računalnih sustava**
- 5. Automatizirano upravljanje temeljem istinitih formula
- 6. …

Formalna verifikacija računalnih sustava

1. Deduktivni pristup

Opis sustava (**implementacija**) dana skupom formula Γ.

Treba dokazati da je formula φ (**specifikacija** sustava) **logička posljedica** skupa Γ.

$\Gamma \models \varphi$ dokaži da su **svi modeli** Γ ujedno i modeli φ

ili

$\Gamma \vdash_L \varphi$ **dokaži** (izvedi) φ uporabom skupa pravila L

(simboličko izvođenje programa)

Obilježja:

- Problem predstavljanja.
- Zahtijeva stručno vođenje (strategije, jednakost, …).
- Primjena ograničena na Ulazno/Izlazne sustave (terminirajuće).
- Može se koristiti za sustave s beskonačnim brojem stanja.

2. Provjera modela (engl. "model checking")

$\Gamma \models \varphi$ provjeravamo **zadovoljivost**

(da li je **φ=neko_ponašanje** istinito **u jednom** modelu

Γ-implementacije, t.j. za jednu interpretaciju)

Obilježja:

Ograničeno na modele s konačnim brojem stanja **= FSM**.

Primjena u reaktivnim sustavima (neterminirajućim).

Automatizirano izvođenje.

Pazi :

Višestruka semantika ("overloading") znaka $\models$:

logička posljedica (uporaba u formalnoj logici) i **zadovoljivost** (uporaba u provjeri modela) - treba uvijek navesti kontekst u kojem se koristi ova oznaka.

Modalna i vremenska (temporalna) 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 propopzicijski simbol

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

Tada:

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

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

Klasifikacija modalne 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 (TL) - višestruki pogledi (1):

TL propozicijska:

Klasična propozicijska logika proširena vremenskim operatorima.

Najviša razina apstrakcije u rasuđivanju.

TL predikatna prvoga reda (varijable, funkcije, predikati, kvantifikatori):

Različiti tipovi TL prvoga reda

interpretirana-neinterpretirana (pretpostavlja ili ne strukturu),

globalne i lokalne varijable,

kvantifikacija preko vremenskih operatora ili ne.

Vremenska logika (TL) - višestruki pogledi (2):

Globalna ili modularna:

Endogena i egzogena. Rasuđivanje o kompletnom sustavu ili ne.

TL linearnog vremena:

U svakom trenutku postoji samo jedan budući trenutak (jedna vremenska crta).

TL 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 TL obuhvaća oba vremena. U digitalnim sustavima uobičajeni su samo operatori budućeg vremena.

Odabiremo**: propozicijska, globalna, grananje, buduće vrijeme**

Model dijela sustava koji želimo formalno provjeriti (verificirati) = model implementacije (*I*)

= Kripke struktura M (Saul Kripke, 1971)

***I* = M = (S, R, L)**

gdje je **:**

S : **konačan** skup stanja

$R \subseteq S \times S$: totalna binarna relacija (svako stanje je obuhvaćeno; ima sljedbenika), t.j.

$\forall s \in S \ (\exists t \in S \mid (s, t) \in R)$

L : funkcija označavanja stanja: $S \rightarrow 2^{AP}$

AP: skup atomičkih propozicijskih simbola

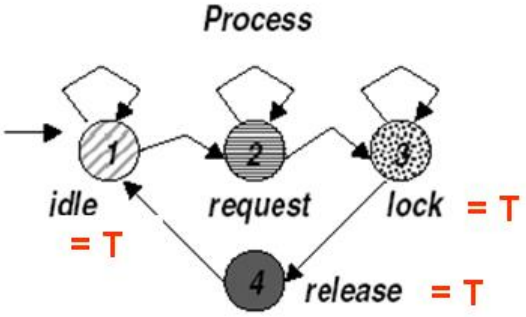
L: (engl. *labeling*) daje interpretaciju svih simbola iz skupa AP za stanje s.

Analogno i formalno jednako modelu automata (**stroju stanja**).

Primjer Kripke strukture

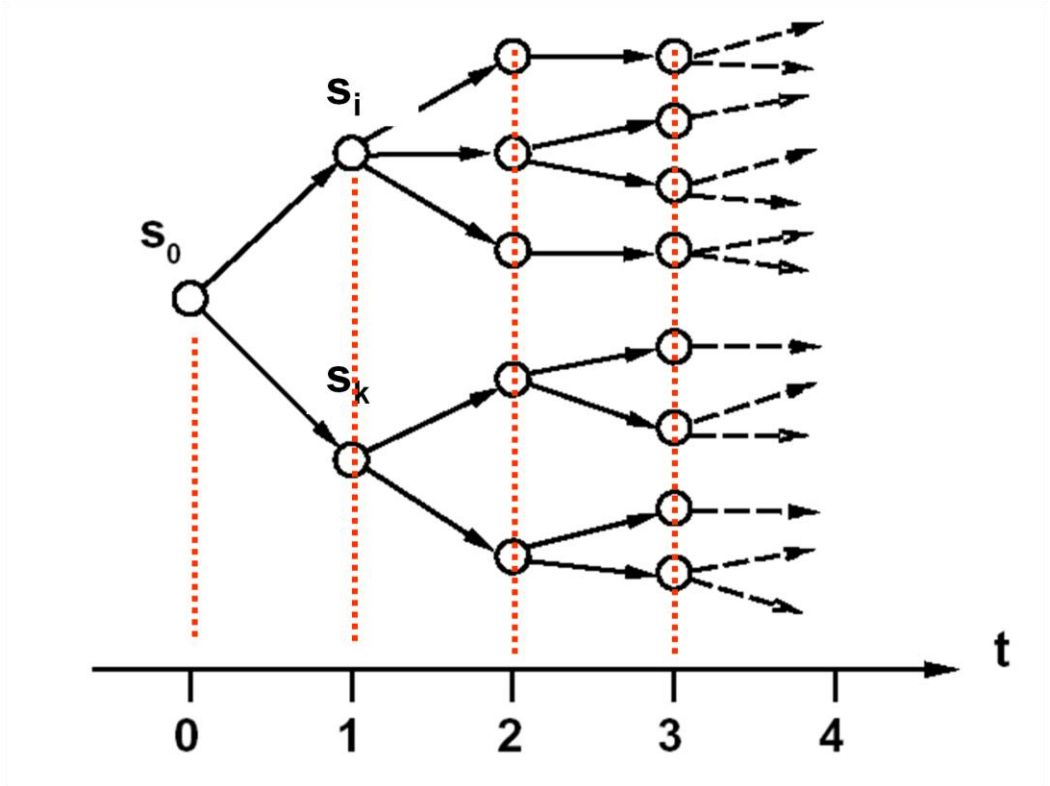
- **Model: Kripke Structure K = (S, R, L)**
  - S: Set of states
  - R: Total Relation on states (transitions)
  - L: Labeling of atomic propositions that are true in a state

- **Example: mutual exclusion for critical section**



$S = \{ 1, 2, 3, 4\}$
$R = \{ (1,1), (1, 2), (2, 2), (2, 3), (3, 3), (3, 4), (4, 1) \}$
$L: L(1) = \{ idle \}$
$L(2) = \{ request \}$
$L(3) = \{ lock \}$
$L(4) = \{ release \}$

Neka definirana Kripke struktura može se promatrati **kao beskonačno stablo izvođenja** sustava ("odmota" se počevši od promatranog stanja s_0). Prikazuju se svi mogući putovi izvođenja sustava.



To je vremenska logika s grananjem Computation Tree Logic = CTL

CTL vremenski operatori (u kontekstu "odmotane" Kripke strukture):

EX	AX
EG	AG
EF	AF
EU	AU

PAZI:
Uvijek u paru !!

CTL sintaksa

AU, EU - binarni operatori

ostali - unarni operatori

Neke **ispravno definirane** CTL formule:

AG (q $\Rightarrow$ EG r)

EG p

E (p U q)

A (p U EFp)

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

Neke **krivo definirane** formule CTL formule:

FG p ; Nema E ili A, t.j. F i G slijede iza E ili A

EF (r U q) ; U se može upariti samo sa A ili E

; Npr. OK je: EF E(r U q), EF A(r U q)

A[p U (q U r)] ; Npr. OK je A[p U E(q U r)]

AF [(r U q) $\wedge$ (p U r)] ; Ispravno je: A(α U β)

; Npr.: A [(p $\wedge$ q) U ($\neg$r $\Rightarrow$ q)]

; ili AF [A(r U q) $\wedge$E(p U r)]

CTL semantika

M = (S, R, L) - model sustava (Kripke struktura)

M, s $\models$ φ formula vrem. Logike φ **je istinita** u modelu M za stanje s

M, s $\not\models$ φ 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 $\models \varphi_2$ (implikacija)

5. M, s $\models$ AX φ akko za sve s_i takve da s $\rightarrow s_i$

vrijedi M, s_i $\models \varphi$ **(u svakom slijedećem stanju)**

6. M, s $\models$ EX φ ako za neki s_i takav da s $\rightarrow s_i$,vrijedi M, s_i $\models \varphi$

(u nekom slijedećem stanju)

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

gdje s = s_1 i 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 $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow$ …, 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 $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow$ …, gdje s = s_1 , postoji neki s_i duž puta

takav da 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 $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow$ …, 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 $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow$ …, 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 je $s = s_1$, 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.

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 i slično je nedjeljivo, t.j. E $\neg$G nije ispravna CTL formula !

Često 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 (adequate set). Postoji više adekvatnih skupova.

Primjeri preslikavanja prirodnog jezika u CTL

(specifikacija ponašanja - *S*)

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 (npr. 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 "send" onda konačno "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:

$M, S_0 \models \phi$, t.j. $\forall s_0 \in S_0$ $M, s_0 \models \phi$ (za svako stanje iz S_0)

Postupak:

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

CTL provjera modela povlači manipulaciju skupovima stanja.

Formalna verifikacija – 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 specificirano ponašanje.
- Programski produkti imaju ogroman skup stanja, pa je moguća provjera samo manjih (kritičnih) dijelova.