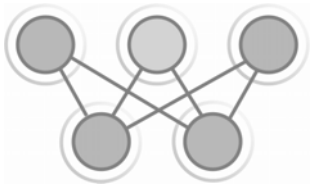
Prof.dr.sc. Bojana Dalbelo Bašić

Fakultet elektrotehnike i računarstva
Zavod za elekroniku, mikroelektroniku, računalne i inteligentne sustave

www.zemris.fer.hr/~bojana
bojana.dalbelo@fer.hr

Sustavi temeljeni na pravilima



ZAKLJUČIVANJE U SUSTAVU TEMELJENOM NA PRAVILIMA

- Rana faza razvoja AI (50-tih i 60-tih godina) sastojao se u pokušaju razvoja sofisticiranih **tehnika zaključivanja**
- Te se tehnike nisu oslanjale na znanje. Cilj u tom periodu bio je razviti programske sustave za opće rješavanje problema (GPS)
- Neuspjeh razvoja općeg sustava i takvog načina rješavanja problema je doveo do drugog pristupa, tzv. razvoja **sustava temeljenih na znanju**

ZAKLJUČIVANJE U SUSTAVU TEMELJENOM NA PRAVILIMA

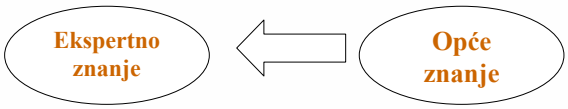
Kako najlakše možemo izraziti ljudsko znanje?

- Velika većina ljudskog znanja (ekspertnog znanja) može se oblikovati kao ***ako – onda* pravilima**
- **Ako** je temperatura pacijenta veća od 38°C **onda** treba prepisati lijek za snižavanje temperature
- **Ako** je svjetlo na semaforu crveno **onda** se zaustavi
- Takva se pravila nazivaju **produkcijska pravila**

Produkcijski sustavi (engl. *production systems*)

ZAKLJUČIVANJE U SUSTAVU TEMELJENOM NA PRAVILIMA

- Važan korak u uspješnom rješavanju problema AI → **redukcija područja ili domene problema**

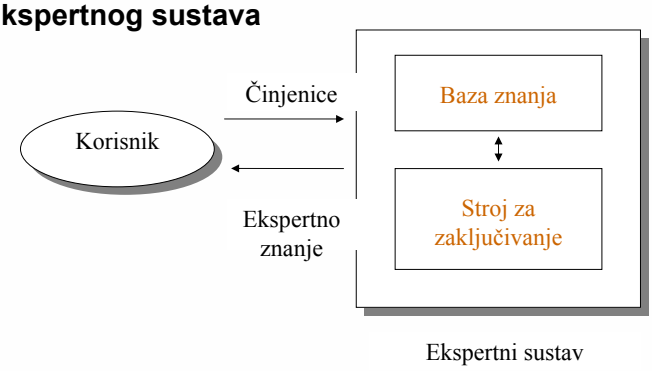


- **Ekspertno znanje** (područno znanje) je specifično znanje koje se odnosi na određeno usko područje, domenu (medicinu, financije, šah itd.), za razliku od općeg znanja rješavanja problema

Ekspertni sustavi (engl. *expert systems*)

ZAKLJUČIVANJE U SUSTAVU TEMELJENOM NA PRAVILIMA

- **Ekspertni sustavi** čine granu AI koja koristi specijalizirano znanje iz neke problemske domene da bi riješila problem na razini ljudskog eksperta
- **Osnovna shema sustava temeljenog na znanju - ekspertnog sustava**



PRODUKCIJSKI SUSTAVI

Začetak produkcijskih sustava:

- Ideja logičara E. Posta 1943. – predložio produkcijska pravila u okviru formalne teorije računarstva. Snaga pp ekvivalentna Turingovom stroju: produkcijski sustavi mogu se koristiti za sve rješive probleme
- 1972. Rad «**Human problem Solving**», Newell i Simon, 920 stranica. Ljudska spoznaja može biti oblikovana ako onda pravilima. Jedno pravilo – granula znanja (engl. *chunk of knowledge*)

PRODUKCIJSKI SUSTAVI

- Model ljudskog spoznajnog procesa se oponaša u prod. sustavu
- Osjeti stimuliraju mozak podražajima. Takvi podražaju aktiviraju znanje u našoj **dugotrajnoj memoriji** (u prod. sustavu to su ako – onda pravila)
- **Kratkotrajna memorija** – privremena pohrana znanja za vrijeme postupka zaključivanja – predstavlja broj «granula znanja» koje mogu biti simultano razmatrane - (pravila koja mogu biti istodobno aktivirana)

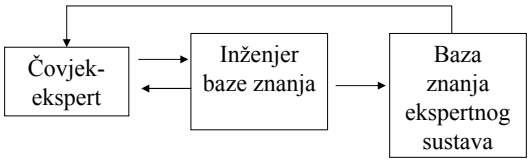
PRODUKCIJSKI SUSTAVI

- **Spoznajni proces** - pronalazi pravila koja će biti aktivirana podražajima – (u prod. sustavu odgovara mu stroj za zaključivanje, on pronalazi pravila, rješava konflikte)
- Newell i Simon – postavili osnovni model modernih produkcijskih sustava
- 1970-tih široko prihvaćena paradigma sustava temeljenih na znanju

PRODUKCIJSKI SUSTAVI

Sinonimi:

- Sustavi temeljeni na pravilima (engl. *rule based systems*)
- Produkcijski sustavi (engl. *production systems*),
- Ekspertni sustavi (engl. *expert systems*) (U klasičnom smislu to su sustavi koji sadrže znanje eksperta dobiveno tijekom niza razgovora **inženjera baze znanja** s ekspertom)



Ovaj postupak je "usko grlo" u izgradnji ekspertnog sustava

PRODUKCIJSKI SUSTAVI

U širem se smislu upotrebljava se izraz:

- Sustavi temeljeni na znanju (engl. *knowledge based systems*)

Nije takav stav o terminima jedinstven:

- "*AI sustavi koji poduprti kvantumom znanja dosežu nivo visoke stručnosti (ekspertnosti) u rješavanju problema nazivaju se* ***sustavi temeljeni na znanju ili ekspertni sustavi****. Izraz ekspertni sustavi rezervirana je za sustave čija baza znanja sadrži znanje ljudskih eksperata, za razliku od znanja nestručnjaka ili znanja sadržanog u udžbenicima. Češće da nego ne, ova dva termina upotrebljavaju se kao sinonimi*"

Edward Feigenbaum,
Stanford University, 1988

PRODUKCIJSKI SUSTAVI

- Produkcijski sustavi - alternativa konvencionalnom algoritamskom programiranju

Područja primjene ekspertnih sustava:

- različiti tipovi medicinskih dijagnoza (interna medicina, pulmologija, infektivne krvne bolesti, itd.)
- dijagnoze složenih elektroničkih i elektromehaničkih uređaja
- mnogobrojne primjene u pravnim sustavima (npr. planiranje prihoda i imovine u svrhu minimalnih poreza)
- primjene u financijama i bankarstvu (npr. evaluacija kreditne sposobnosti tvrtke ili pojedinca)

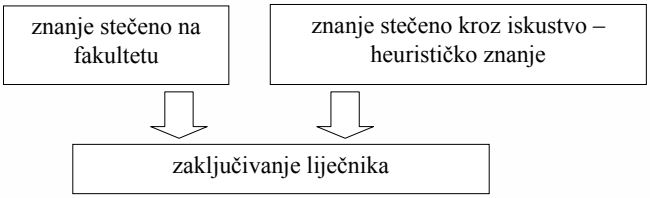
PRODUKCIJSKI SUSTAVI

Područja primjene ekspertnih sustava:

- planiranje eksperimenta u biologiji, kemiji, molekularnoj genetici
- optimalna konfiguracija komponenata kompleksnih sustava koji moraju zadovoljavati određene uvjete
- dizajn VLSI sustava
- razne primjene u vojne svrhe: planiranje snabdijevanja jedinica do nadzora oceana
- edukativne svrhe – *tutoring* sustavi

PRODUKCIJSKI SUSTAVI

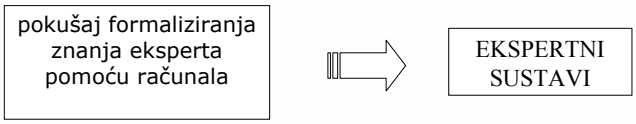
- Primjer područja primjene: medicina
- Postavljanje dijagnoze i određivanje terapije na temelju simptoma



Znanje eksperta – ljudsko znanje:

- čovjek – ekspert, stručnjak
- utjecaj umora ili emocija na stručnu procjenu
- vijek trajanja znanja ograničen

PRODUKCIJSKI SUSTAVI



- Posebna (ekspertna) znanja prikupljena i organizirana za posebnu (usko stručnu) namjensku uporabu

Primjeri ranih uspješnih ekspertnih sustava:

- **DENDRAL** –otkrivanje molekularne strukture tvari na temelju spektrometrije.
- **MYCIN** – dijagnosticiranje infektivnih krvnih bolesti
- **PROSPECTOR** – procjenjivanje postojanja ruda na temelju zemljopisnih karakteristika
- **XCON/R1 – DEC** sustav za izbor komponenata za oblikovanje kompleksnih računalnih sustava

PRODUKCIJSKI SUSTAVI

Ekspertni sustavi – prednosti:

- izgrađeni na temelju znanja više eksperata - repozitorij ekspertnog znanja za neko područje
- mogu se umnožiti i biti u širokoj uporabi

Sustavi temeljeni na pravilima temeljeni su na:

- **produkcijskim pravilima** (engl. *production rules, prodrules*)

Ako ..... **onda** .....

uvjet, stanje (engl.*condition*) premisa, antecedens

akcija, posljedica (engl. *action*) zaključak, konsekvens

PRODUKCIJSKI SUSTAVI

- **Ako** pada kiša **onda** ponesi kišobran
- **Ako** se motor automobila ne može upaliti i nema svjetla
  onda je problem u akumulatoru ili u kablovima

uvjet-akcija

- **Ako** je pacijent kronično lošeg općeg stanja i
  spol pacijenta je ženski i
  godine starosti < 30 i
  pacijent ima simptom A i
  biokemijski test pokazuje vrijednost C
- **onda** je dijagnoza autoimuni kronični hepatitis

premisa - zaključak

PRODUKCIJSKI SUSTAVI

- **Ako** je Ivan student FER-a u Zagrebu
  onda je Ivan student zagrebačkog Sveučilišta

stanje-posljedica

Sustavi temeljeni na pravilima vrlo su često korišteni jer se većina ljudskog znanja može izraziti produkcijskim pravilima (ako-onda).

PRODUKCIJSKI SUSTAVI

- U logici: implikacija **A** $\rightarrow$ **B** (Ako vrijedi A onda vrijedi B)

logičke formule

- U produkcijskim pravilima: **Ako A onda B**

akcije, naredbe, jednadžbe, programi…

- Produkcijski sustavi su kombinacija dviju vrsti znanja:

[1] činjeničnog (A je istinito) **A**

[2] uvjetnog (vrijedi pravilo) **A** $\rightarrow$ **B**

(B je istinito) **B**

MODUS PONENS

PRODUKCIJSKI SUSTAVI

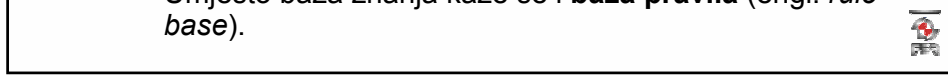
Definicija - **PRODUKCIJSKI SUSTAV** definiran je sa:

1. **SKUPOM PRODUKCIJSKIH PRAVILA** (produkcija je par uvjet-akcija) [2]
2. **RADNOM MEMORIJOM** (baza činjenica koja sadrži opis tekućeg stanja svijeta u postupku zaključivanja) [1]

BAZA ZNANJA

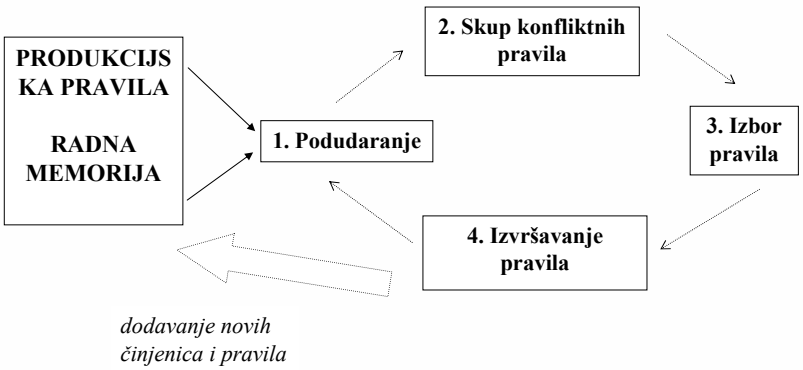
3. **CIKLUSOM PODUDARANJE-DJELOVANJE**
   To je upravljački mahanizam koji se sastoji od slijeda akcija: podudaranje uzoraka (engl. *pattern matching*) – rješavanje konflikata izborom pravila - izvršavanje pravila

- U užem smislu se naziv **baza znanja** (engl. *knowledge base*) upotrebljava za skup produkcijskih pravila. Umjesto baza znanja kaže se i **baza pravila** (engl. *rule base*).



PRODUKCIJSKI SUSTAVI

- Shema rada produkcijskog sustava



PRODUKCIJSKI SUSTAVI

Opis produkcijskog sustava

1. SKUP PRODUKCIJSKIH PRAVILA – produkcijsko pravilo (još se naziva *uvjet-akcija* par) predstavlja jedan djelić znanja iz problemske domene. Uvjetni dio pravila je uzorak koji određuje u kojem se slučaju pravilo primijenjuje

2. RADNA MEMORIJA sadrži skup činjenica. Čnjenice (uzorci) iz radne memorije predstavljaju trenutno stanje u postupku rješavanja problema. Uzorci se uspoređuju s uvjetnim dijelovima pravila

PRODUKCIJSKI SUSTAVI

3. CIKLUS PODUDARANJE-DJELOVANJE – **je upravljačka struktura produkcijskog sustava**.

   Uzorci iz radne memorije uspoređuju se s uvjetnim dijelovima pravila iz baze pravila.

   Kada se uvjetni dio nekog pravila **podudara** s uzorcima iz radne memorije, onda je to pravilo omogućeno i ono se izdvaja u **skup konfliktnih pravila**. Na početku ciklusa raspoznavanja konfliktni skup je prazan.

PRODUKCIJSKI SUSTAVI

Nakon postupka podudaranja konfliktni skup može sadržavati jedno ili više omogućenih pravila.

Ako je skup konfliktnih pravila prazan onda je postupak završen,

Ako ima više omogućenih pravila izabire se JEDNO pravilo koje se izvršava, tj. koje pali.

Odabir pravila koje pali naziva se **razrješavanje konflikata.** Paljenje pravila znači izvršavanje desnog dijela odabranog pravila koje mijenja sadržaj radne memorije.

Nakon paljenja pravila upravljački ciklus ponavlja se s promijenjenim sadržajem radne memorije

PRODUKCIJSKI SUSTAVI

- **Razrješavanje konflikata** može biti jednostavno – npr. pali se prvo omogućeno pravilo, ali može biti i složeno, tj. može uključivati heuristiku za izbor pravila koje pali
- *Napomena*: Kada se neko produkcijsko *pravilo izvršava* još se kaže da *pravilo pali*

Primjer

- Jednostavan produkcijski sustav za sortiranje stringova koji se sastoje od slova **a**, **b** i **c**

SKUP PRODUKCIJSKIH PRAVILA:

1. **ba → ab**
2. **ca → ac**
3. **cb → bc**

PRODUKCIJSKI SUSTAVI

1. ba → ab
2. ca → ac
3. cb → bc

Iteracija br.	RADNA MEMORIJA	Podudaranje	Skup konfliktnih pravila	Izbor pravila koje pali
0	**cbaca**		**3, 1, 2**	**1**
1	**cabca**		**2**	**2**
2	**acbca**		**3, 2**	**2**
3	**acbac**		**1, 3**	**1**
4	**acabc**		**2**	**2**
5	**aacbc**		**3**	**3**
6	**aabcc**		∅	**Stop**

PRODUKCIJSKI SUSTAVI

1. Produkcijsko pravilo je u konfliktnom skupu ako se uvjetni dio pravila podudara sa stringom u radnoj memoriji
2. Pravilo koje pali bira se prema prioritetu (redni broj pravila)
3. Paljenjem pravila, podniz u radnoj memoriji koji se podudara s podnizom iz uvjetnog dijela pravila zamijenjen je podstringom desnog dijela pravila

Napomena

- Ako je produkcijsko pravilo u konfliktnom skupu kaže se da je **omogućeno**

PRODUKCIJSKI SUSTAVI

Dodavanje novog znanja u postojeću bazu znanja

Primjer:

- **Pravila**

  [Pravilo 1] **Ako** je temperatura iznad 30°C
  onda vrijeme je vruće

  [Pravilo 2] **Ako** je relativna vlažnost veća od 65%
  onda atmosfera je vlažna

  [Pravilo 3] **Ako** je vrijeme vruće i atmosfera vlažna
  onda postoji vjerojatnost oluje

Činjenice

- Temperatura je 35°C → Temperatura iznad 30°C
  Vlaga je 70% → Relativna vlažnost veća od 65%

Nova činjenica

- Postoji vjerojatnost oluje

PRODUKCIJSKI SUSTAVI

Produkcijski sustavi:

1) su nemonotoni (engl. *nonmonotonic*)
2) prihvaćaju određen stupanj netočnosti

MONOTONA LOGIKA (propozicijska logika i FOPL)

- Zaključci dobiveni valjanim zaključivanjem ostaju valjani. Pridodavanjem novog znanja (činjenica) povećava se količina znanja dok valjanost starog znanja ostaje ista

NEMONOTONA LOGIKA

- Nove činjenice koje se pridodaju bazi znanja mogu biti u kontradikciji s postojećim znanjem. Time se obezvređuje staro znanje

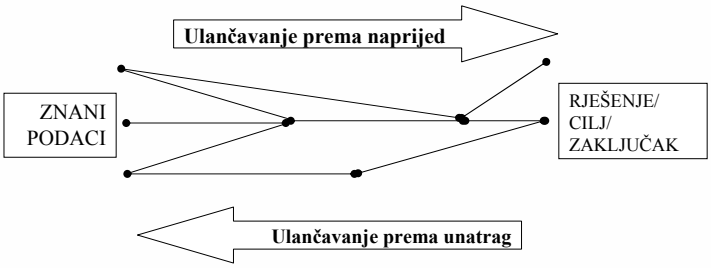
PRODUKCIJSKI SUSTAVI

Primjer

- Pretpostavimo da se baza znanja sastoji od
  - P
  - P→ Q
- Pomoću modus ponensa zaključujemo Q.
- Taj se zaključak dodaje bazi znanja.
- Sada pretpostavimo da je u kasnijem postupku djelovanja sustava uvedena činjenica ~P.
- Pri dodavanju takve činjenice bazi znanja potrebno je ukloniti P i Q.
- Tu funkciju obavlja sustav za održavanje istinitosti – TMS (engl. *Truth Maintenance System)*

PROCES ZAKLJUČIVANJA U SUSTAVU TEMELJENOM NA PRAVILIMA

- Niz sastavljen od višestrukih zaključaka koji povezuju dani početni opis problema s rješenjem naziva se **LANAC**



- Postupak zaključivanja, tj. automatsko napredovanje kroz lanac, naziva se ulančavanje

PROCES ZAKLJUČIVANJA U SUSTAVU TEMELJENOM NA PRAVILIMA

Postoje dva glavna načina napredovanja prema zaključcima:

1. **ULANČAVANJE PRAVILA PREMA NAPRIJED**

- → započinjanje sa znanim podacima i napredovanje prema zaključku (*engl.* ***forward chaining (forchaining), data driven processing***, event driven, bottom-up, antecedent, pattern directed **processing ⇔ reasoning)**

2. **ULANČAVANJE PRAVILA UNATRAG**

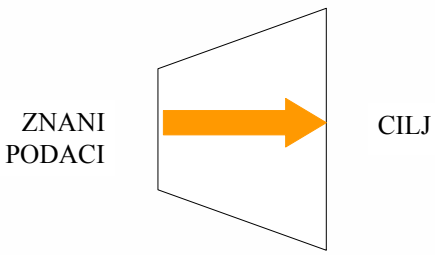
- → izbor mogućeg zaključka (hipoteza) i pokušaj dokazivanja valjanosti hipoteze traženjem valjanih potpora (dokaza, engl. *evidence*).

  (engl. ***backward chaining (backchaining), goal driven processing***, goal driven, top-down, consequent, expectation driven processing)

PROCES ZAKLJUČIVANJA U SUSTAVU TEMELJENOM NA PRAVILIMA

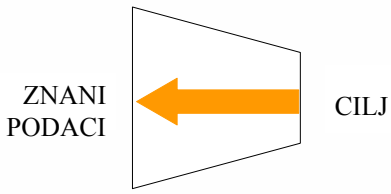
- Ulančavanje prema naprijed - kada ima malo podataka i puno mogućih rješenja

  (za problemske domene koje uključuju sintezu: za dizajniranje, planiranje, raspoređivanje, za nadzor i dijagnostiku sustava za rad u stvarnim vremenu)



PROCES ZAKLJUČIVANJA U SUSTAVU TEMELJENOM NA PRAVILIMA

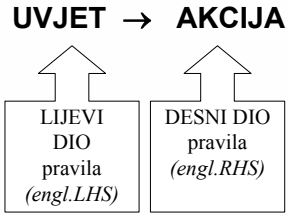
- Ulančavanje unatrag razuman je izbor kada je malo mogućih zaključaka/ciljeva i puno znanih podataka. (Za probleme dijagnosticiranja, klasificiranja)



- Izbor metode zaključivanja ovisi o osobinama problemske domene i o načinu zaključivanja eksperta
- Moguće je implementirati i obostrano (bidirekcionalno) zaključivanje

PROCES ZAKLJUČIVANJA U SUSTAVU TEMELJENOM NA PRAVILIMA

- **Produkcijsko pravilo:**



ULANČAVANJE PREMA NAPRIJED

- Zaključivanje prema naprijed započinje znanim podacima, inicijalnim **opisom problema** (npr. skupom logičkih aksioma, simptomima bolesti, podacima koji trebaju biti protumačeni itd.) koji je pohranjen u radnoj memoriji.
- Provjerava se svako pravilo da li dani podaci zadovoljavaju premise pravila. Ako je pravilo zadovoljeno, ono može biti izvršeno, i izvedeni su novi podaci koji mogu biti korišteni za zadovoljavanje drugih pravila. Postupak provjere naziva se interpretacija pravila.

Nazivi

- postupak interpretacije pravila ≡ zaključivanje prema naprijed ≡ ulančavanje prema naprijed

ULANČAVANJE PREMA NAPRIJED

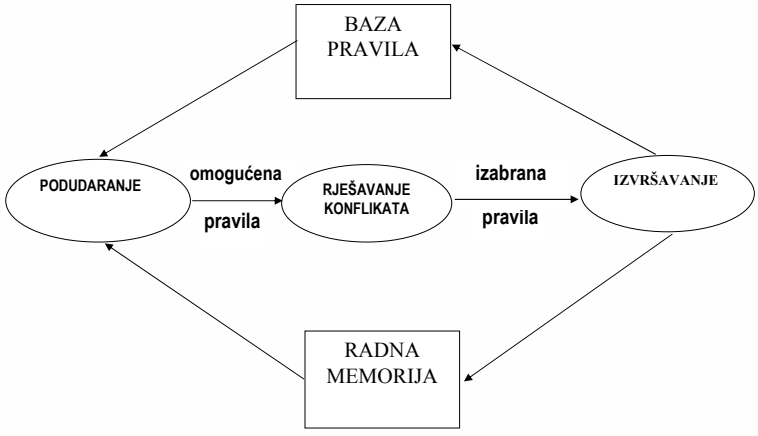
STROJ ZA ZAKLJUČIVANJE (engl. *inference engine*) je upravljački mahanizam koji u postupku ulančavanja prema naprijed izvodi sljedeće korake:

1. **Podudaranje** - upravljački ciklus započinje podudaranjem stanja u radnoj memoriji s **LIJEVIM** dijelom produkcijskog pravila
2. **Razrješavanje konflikata** – ako je tijekom podudaranja nađeno više pravila koja su omogućena – izabire se pravilo najvećeg prioriteta

ULANČAVANJE PREMA NAPRIJED

3. **Izvršavanje** (paljenje) pravila – izvršavanje desnog dijela (akcija) produkcijskog pravila rezultira:
   - novom činjenicom koja je dodana u radnu memoriju (novo tekuće stanje svijeta, ili engl. *current state of the world*), ili
   - novim pravilom koje je dodano u bazu znanja i može biti razmatrano za izvršavanje u sljedećim ciklusima

ULANČAVANJE PREMA NAPRIJED



ULANČAVANJE PREMA NAPRIJED

Primjer

- Baza pravila za određivanje vrsta voća
- Parametri (tj. varijable) i njihove vrijednosti su:

Oblik:	izdužen		Promjer:	> 10 cm
	okrugli			< 10 cm
	zaoblen		Vrsta_voćke:	loza
Površina:	glatka			stablo
	hrapava		Voće:	banana
Boja:	zelena			lubenica
	žuta			dinja
	žuto-smeđa			kanalupe
	crvena			jabuka
	plava			marelica
	narančasta			višnja
Broj_sjemenki	> 1			breskva
	= 1			šljiva
Vrsta_sjemenke	višestruke			naranča
	koštunjasta			

ULANČAVANJE PREMA NAPRIJED

PRAVILA

1	AKO	Oblik = izdužen & Boja = zelena ili žuta
	ONDA	Voće = banana
2	AKO	Oblik = okrugli ili zaobljen & Promjer > 10 cm
	ONDA	Vrsta_voćke = loza
3	AKO	Oblik = okrugli & Promjer < 10 cm
	ONDA	Vrsta_voćke = stablo
4	AKO	Broj_sjemenki = 1
	ONDA	Vrsta_sjemenke = koštunjasta
5	AKO	Broj_sjemenki > 1
	ONDA	Vrsta_sjemenke = višestruke
6	AKO	Vrsta_voćke = loza & Boja = zelena
	ONDA	Voće = lubenica

ULANČAVANJE PREMA NAPRIJED

7	AKO	Vrsta_voćke = loza & Površina = glatka & Boja = žuta
	ONDA	Voće = dinja
8	AKO	Vrsta_voćke = loza & Površina = hrapava & Boja = žuto-smeđa
	ONDA	Voće = kantalupe
9	AKO	Vrsta_voćke = stablo & Boja = narančasta & Vrsta_sjemenke = koštunjasta
	ONDA	Voće = marelica
10	AKO	Vrsta_voćke = stablo & Boja = narančasta & Vrsta _sjemenke = višestruke
	ONDA	Voće = naranča
11	AKO	Vrsta_voćke = stablo & Boja = crvena & Vrsta_sjemenke = koštunjasta
	ONDA	Voće = višnja
12	AKO	Vrsta_voćke = stablo & Boja = narančasta & Vrsta _sjemenke = koštunjasta
	ONDA	Voće = breskva
13	AKO	Vrsta_voćke = stablo & Boja = žuta ili zelena ili crvena & Vrsta _sjemenke = višestruke
	ONDA	Voće = jabuka
14	AKO	Vrsta_voćke = stablo & Boja = plava & Vrsta _sjemenke = koštunjasta
	ONDA	Voće = šljiva

ULANČAVANJE PREMA NAPRIJED

- Znani podaci: Promjer = 2 cm
  Oblik = okrugli
  Broj_sjemenki = 1
  Boja = crvena

Strategija izbora pravila: pravilo s najmanjim brojem

	Radna memorija	*Skup konfliktnih pravila*	*Pravilo koje pali*
0	Promjer = 2 cm Oblik = okrugli Broj_sjemenki = 1 Boja = crvena	3,4	3
1	Vrsta_voćke: stablo	3, 4	4
2	Vrsta_sjemenke=koštunjasta	3, 4, 11	11
3	Voće = višnja	3, 4, 11	STOP

ULANČAVANJE PREMA NAPRIJED

- U 3. koraku nema više novih omogućenih pravila – izvršavanje se zaustavlja

Napomena

- Uočite da se u postupku podudaranja pravila ispituje u bazi znanja da li određeni parametar ima pridjeljenu vrijednost koja je uvjet u tom pravilu

RETE ALGORITAM

- **(Forgy**, C. L. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem, *Artificial Intelligence*, 1982.) – RETE razvijen u okviru ljuske OPS-5

- Ciklus podudaranje-razrješavanje konflikata—izvršavanje podrazumijeva da se sva pravila uparuju sa svim činjenicama u radnoj memoriji kako bi se odredio skup konfliktnih pravila ⇒ to je neučinkovito!

RETE ALGORITAM

- Neka sustav ima:

r - *pravila*
f - *činjenica*
p - *premisa*

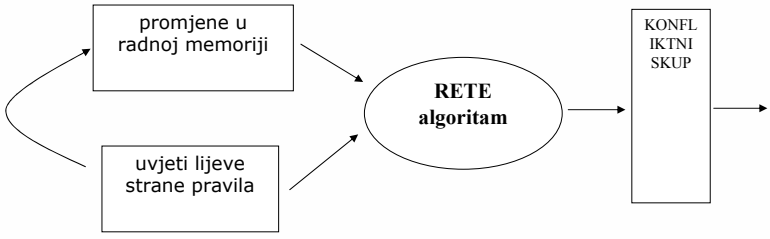
$r * f^p$ usporedbi u svakom ciklusu

- Na primjer: Prosječni broj pravila je 150
  Prosječni broj premisa po pravilu 4
  Prosječni broj činjenica 20
  $150*20^4 = 24{,}000{,}000$ usporedbi

RETE ALGORITAM

Pretpostavke RETE algoritma:

1. Sadržaj radne memorije ne mijenja se bitno iz ciklusa u ciklus. Takva se ustrajnost podataka naziva "vremenska redundancija".



RETE ALGORITAM

- Umjesto usporedbe svih pravila sa svim činjenicama u svakom ciklusu da bi se vidjelo koja su pravila zadovoljena, RETE algoritam prati promjene u radnoj memoriji (dodavanja i uklanjanja činjenica) i u skladu s time direktno ažurira skup konfliktnih pravila (pravila koja su zadovoljena).
- Kada neko pravilo pali, uklanja se iz konfliktnog skupa dok su ostala pravila sačuvana za sljedeći ciklus.
- Indeksiranjem pravila s uvjetnim izrazima koja se pojavljuju na lijevoj strani pravila, ispituju se samo ona pravila koja se mogu podudarati sa sadržajem radne memorije. To bitno reducira broj usporedbi u svakom ciklusu

RETE ALGORITAM

2. Više pravila u radnoj memoriji imaju iste uvjete u lijevom dijelu pravila. Ponavljajuće ispitivanje uvjeta može se izbjeći grupiranjem pravila koja imaju iste uvjete

RETE algoritam zahtijeva razvoj:

- **Mreže uzoraka** (pattern network) – skup stabala koje je izgrađeno od svih premisa svih pravila
- **Mreže udruživanja** (join network) - povezuju listove stabala redoslijedom kako oblikuju klauzulu, i uspoređuju istoimene varijable u svrhu utvrđivanja imaju li iste vrijednosti

SHEME ZA RJEŠAVANJE KONFLIKATA KOD ULANČAVANJA UNAPRIJED

Četiri kategorije shema:

- Prema
  1. Broju pravila koja se izvršavaju
  2. Uređenju pravila
  3. Složenosti pravila
  4. Uređenju podataka

1. Određen broj pravila koja se mogu paliti tijekom jednog ciklusa:
   - jedno pravilo
   - veći broj pravila
   - sva pravila u konfliktnom skupu (krajnji slučaj)

SHEME ZA RJEŠAVANJE KONFLIKATA KOD ULANČAVANJA UNAPRIJED

2. Slijed ili uređenje pravila utječe na odluku:
   a) uvijek pali pravilo s najnižim (slijednim) brojem, ili
   b) pali prvo pravilo koje slijedi pravilu koje je bilo paljeno u prošlom ciklusu, ili
   c) pravilo s najnižim brojem, ali koje daje nove činjenice, itd.

3. Ispitaj složenost pravila (npr. broj premisa u LHS) i izaberi najsloženije pravilo;
   Alternativno: izaberi najopćenitije pravilo (obično ima mali broj premisa)

SHEME ZA RJEŠAVANJE KONFLIKATA KOD ULANČAVANJA UNAPRIJED

4. Ispitaj uređenje podataka i izvrši pravilo koje se podudara s najstarijim podacima u bazi
   Alternativno: … s najnovijim podacima u bazi

Moguće su kombinacije ovih shema

Još neke sheme:

a) Pali pravilo koje je najdulje na listi izvršivih pravila

b) na temelju savjeta eksperta pravilima su dodijeljeni prioriteti – palli pravilo s najvećim prioritetom

SHEME ZA RJEŠAVANJE KONFLIKATA KOD ULANČAVANJA UNAPRIJED

c) identična 3. shemi (složenost pravila)
   primjer: (J) If $e_1 \wedge e_2$ then h_1
   (J') If e_1 then h_2

- Neko produkcijsko pravilo J specifičnije je nego je to pravilo J' akko je slučaj podudaranja LHS od pravila J podskup slučaja podudaranja LHS pravila J'
- U našem primjeru J je specifičnije produkcijsko pravilo od J'.
  primjer: (I) If P(b) then h_3
  (I') If $\forall$xP(x) then h_4

Ovdje je I specifičnije of I' (P predikat; b je konstanta)

SHEME ZA RJEŠAVANJE KONFLIKATA KOD ULANČAVANJA UNAPRIJED

d) Uređivanje slijeda paljenja pravila uporabom meta produkcijskih pravila. To su pravila koja govore o pravilima u bazi znanja, tj. određuju slijed paljenja pravila u konfliktnim situacijama

ULANČAVANJE PRAVILA UNATRAG

Ulančavanje unatrag

- bitno se razlikuje od ulančavanja unaprijed iako oba načina zaključivanja ispituju i primjenjuju pravila
- započinje sa željenim ciljem (hipoteza) i ispituje se da li postojeće činjenice podržavaju izvođenje vrijednosti za taj zaključak
- Sistem započinje praznom bazom činjenica. Zadaje se lista ciljeva za koju sustav pokušava izvesti vrijednosti

ULANČAVANJE PRAVILA UNATRAG

Koraci:

1. Oblikuj stog inicijalno sastavljen od najvažnijih ciljeva (hipoteza) koje treba dokazati

2. Na vrhu stoga je hipoteza koju treba dokazati.
   Ako je stog prazan, onda je KRAJ

3. Izdvoji sva pravila koja mogu zadovoljavati dani cilj (tj. izdvoji sva pravila čija se DESNA strana podudara s ciljem)

ULANČAVANJE PRAVILA UNATRAG

4. Za svako od tih pravila učini redom:

a) **Ako** su sve premise pravila zadovoljene (svaki parametar premise ima vrijednost sadržanu u radnoj memoriji)

tada izvrši pravilo, tj. DESNU stranu tog pravila, tj. dodaj zaključke u radnu memoriju. Ne razmatraj više pravila za taj cilj - vrijednost cilja upravo je izvedena paljenjem tog pravila.

Ako je cilj bio vršni cilj **tada** ukloni cilj sa stoga & vrati se na korak 2.

Ako je cilj bio međucilj **tada** ukloni cilj sa stoga & vrati se privremeno suspendiranom cilju

ULANČAVANJE PRAVILA UNATRAG

b) **Ako** se vrijednost parametra nađena u memoriji ne podudara sa vrijednošću parametra premise

onda ne izvršavaj to pravilo

c) **Ako** premise pravila nisu zadovoljene zato jer jedna od parametarskih vrijednosti te premise nije u radnoj memoriji,

tada potraži pravilo čija desna strana izvodi vrijednost tog parametra.

Ako barem jedno takvo pravilo postoji

tada odredi taj parametar kao podcilj, tj.

postavi taj parametar na vrh stoga &

idi na korak 2

ULANČAVANJE PRAVILA UNATRAG

d) **Ako** korak (c) ne može naći pravilo koje izvodi potrebnu vrijednost tekućeg parametra

tada pitaj korisnika za tu vrijednost parametra & dodaj vrijednost u radnu memoriju.

Idi na korak 4a i razmatraj sljedeću premisu tekućeg pravila

5. **Ako** su sva pravila koja mogu zadovoljavati tekući cilj provjerena i ako ni jedno nije uspjelo izvesti vrijednost cilja

tada cilj ostaje neodređen. Makni cilj sa stoga i prijeđi na korak 2

ULANČAVANJE PRAVILA UNATRAG

Primjer ulančavanja unatrag

- Neka je naša početna hipoteza da se radi o komadu voća
- **hipoteza = (voće).** Imajući na umu da se radi o višnji - slijedimo sada rad sustava ulančavanjem unatrag, da bi vidjeli je li moguće izvesti da je voće višnja
- U drugom koraku izdvajamo SVA pravila koja izvode vrijednost za voće

ULANČAVANJE PRAVILA UNATRAG

Cilj (stog)	**RADNA MEMORIJA**		**KONFLIKTNI SKUP**	**parametar koji se provjerava**
(**voće**)			**1**,6,7,8,9,10,11,12, 13,14	oblik nije u RM i nije na RHS od nekog pravila - **?oblik?**
(voće)	oblik = okrugli	(~~R1~~)	**6**,7,8,9,10,11, 12,13,14	vrsta_voćke –RHS od pravila 2 i 3
(**vrsta_voćke** je trenutni cilj; cilj **voće** je privremeno suspendiran)			2,3,6,7,8,9,10,11, 12,13,14	oblik – nalazi se u RM – OK promjer – nije u RM i nije na RHS ni jednog **pravila ?promjer?**
(vrsta_voćke voće)	promjer < 10 cm	(~~R2~~)	**3-PALI,** 6,7,8,9,10,11, 2,13,14	obje premise od R3 su u RM
(**voće**)	vrsta_voćke= stablo		**6**,7,8,9,10,11, 12,13,14	prva premisa od R6 nije zadovoljena
(voće)		(~~R6~~)	**7**,8,9,10,11, 12,13,14	prva premisa od R7 nije zadovoljena
(voće)		(~~R7~~)	**8**,9,10,11, 12,13,14	prva premisa od R8 nije zadovoljena

ULANČAVANJE PRAVILA UNATRAG

(voće)		(~~R8~~)	**9**,10,11, 12,13,14	vrsta_voćke je u RM; 2. premisa nije RM i nije na RHS od ni jednog pravila **? boja ?**
(**voće**)	boja = crvena	(~~R9~~)	**10**,11, 12,13,14	druga premisa od R10 nije zadovoljena
(**voće**)		(~~R10~~)	11, 12,13,14	
….	…	..	…	…
(**voće**)			**11 - PALI**	voće=višnja

? … ? – upit korisniku. Sva uklonjena pravila sa vrha stoga za koje ne piše da pale nisu bila zadovoljena. Ulančavanje ide unatrag sve dok se desni dio pravila pojavljuje u nekom drugom pravilu na lijevoj strani

PRIMJER ULANČAVANJA PREMA NAPRIJED

Skup produkcijskih pravila:

1. $p \wedge q \rightarrow$ *cilj*
2. $r \wedge s \rightarrow p$
3. $w \wedge r \rightarrow q$
4. $t \wedge u \rightarrow q$
5. $v \rightarrow s$
6. početak $\rightarrow v \wedge r \wedge q$

- Strategija rješavanja konflikata: odaberi pravilo iz skupa konfliktnih pravila koje je prvo palilo ili nije uopće

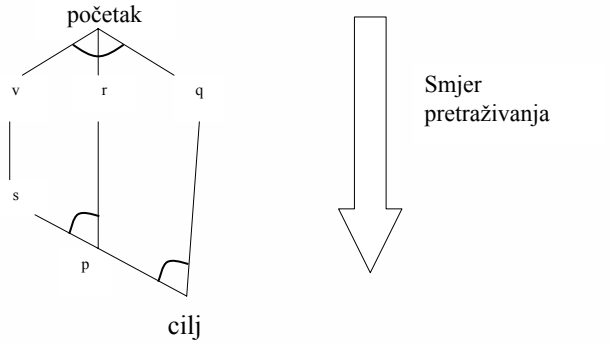
PRIMJER ULANČAVANJA PREMA NAPRIJED

- Izvršavanje

	Radna memorija	Skup konfliktnih pravila	Pravilo koje pali
0	*Početak*	6	6
1	*početak, v, r, q*	6,5	5
2	*početak, v, r, q, s*	6,5,2	2
3	*početak v, r, q, s ,p*	6,5,2,1	1
4	*početak v, r, q, s ,p cilj*	6,5,2,1	STOP

PRIMJER ULANČAVANJA PREMA NAPRIJED

- Prostor koji se pretražuje izvršavanjem



PRIMJER ULANČAVANJA PREMA UNATRAG

Skup produkcijskih pravila

1. $p \wedge q \rightarrow$ *cilj*
2. $r \wedge s \rightarrow p$
3. $w \wedge r \rightarrow q$
4. $t \wedge u \rightarrow q$
5. $v \rightarrow s$
6. početak $\rightarrow v \wedge r \wedge q$

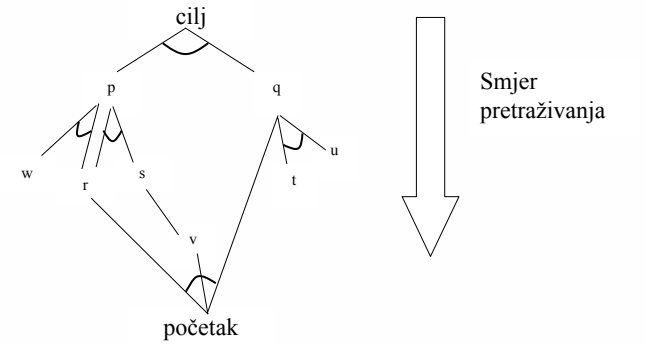
PRIMJER ULANČAVANJA PREMA UNATRAG

- Izvršavanje

CILJ (STOG)	RADNA MEMORIJA	SKUP KONFLIKTNIH PRAVILA	Parametar koji se provjerava
cilj		1	p–nije u RM
p, cilj		2	r–nije u RM
r, p, cilj	početak	6 – PALI	?početak?
p, cilj	početak, v, r, q	2	r je u RM, s nije
s, p, cilj	početak, v, r, q	5 - PALI	v je u RM
p, cilj	početak, v, r, q, s	2 - PALI	r, s – u RM
cilj	početak, v, r, q, s, p	1 – PALI	p, q u RM
	početak, v, r, q, s, p, cilj	STOP	

PRIMJER ULANČAVANJA PREMA UNATRAG

- Prostor koji se pretražuje izvršavanjem



Pri ulančavanju unatrag sustav traži potporu (dokaze) za cilj i podciljeve. Takvo je pretraživanje pogodno za klasifikaciju i dijagnostiku

SAŽETAK

ulančavanje unatrag – pretraživanje u dubinu
(*engl. depth first search*)

ulančavanje prema naprijed – pretraživanje u širinu
(*engl. breadth first search*)

- MYCIN (Shortliffe, 1976) – primjer ulančavanja unatrag
- XCON/R1 (MCDermott, DEC, 1978) – primjer ulančavanja prema naprijed

OBLICI PRAVILA

Ljuska ekspertnih sustava – alat namijenjen razvoju ekspertnog sustava. Sadrži sve elemente sustava osim znanja.

- CLIPS (**C** **L**anguage **I**ntegrated **P**roduction **S**ystems – razvijen u NASA/Johnson Space Center). Kasnije verzije kompatibilne sa C++
- Java verzija CLIPSa **JESS**
- OPS5 (korišten za XCON)
- OPS83
- ART
- KEE (IntelliCorp, 1983)
- EMYCIN (prazan MYCIN)

OBLICI PRAVILA

- Sintaksa jezika za razvoj produkcijskih sustava vrlo je različita
- Dva su osnovna sintaktička oblika za informacije pohranjene u radnoj memoriji:

1. **(OBJEKT, ATRIBUT, VRIJEDNOST)**

Primjer

(OBJEKT, ATRIBUT, VRIJEDNOST) može na primjer biti (moj_auto, boja, crvena)

- OPS5 dozvoljava bilo koji broj atributa i vrijednosti pridruženih jednom objektu (to jača izražajnu moć jezika)
- EMYCIN dozvoljava osnovnu trojku s dodanim faktorom sigurnosti (engl. *certainty factor*)

OBLICI PRAVILA

2. **(ATRIBUT, RELACIJA, VRIJEDNOST)**

Primjer

(ATRIBUT - RELACIJA - VRIJEDNOST) može na primjer biti (TEMPERATURA VEĆA_OD 20)

Produkcijski sustavi (PS) razlikuju se i u dozvoljenim oblicima pravila koja čine bazu pravila

Najčešće razlike su u:

a) akcijama koje su dozvoljene na desnoj strani,
b) broju premisa koje su dozoljene na lijevoj strani, i
c) mogućnosti pojavljivanja varijabli na obje strane pravila

OBLICI PRAVILA

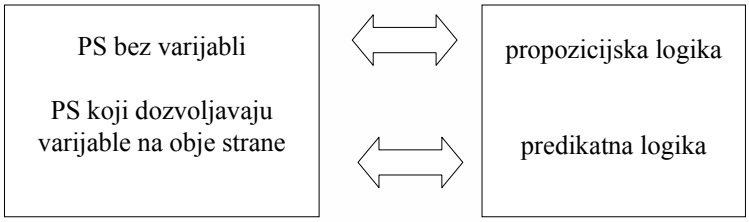
Komentar za (a):

- Najjednostavniji sustavi - tip akcije koja se izvodi paljenjem pravila je dodavanje činjenica u bazu znanja ili njihova modifikacija,
- OPS5 dozvoljava akcije poput izvođenja proizvoljnog koda (Sjeti se razlike između implikacije u FOPL i produkcijskog pravila!)

Komentar za(c):

- Najjednostavniji sustavi – nema varijabli ni na jednoj strani → vrlo slaba moć oblikovanja znanja → premisa na LHS mora biti identična uzorku u RM
- Na primjer, takvi sustavi ne mogu oblikovati pravilo:
  Za svaki auto vrijedi da ako je akumulator prazan tada se auto ne može pokrenuti
- Većina PS dozvoljava pravila s varijablama na obje strane

OBLICI PRAVILA



- Podudaranje uzorka na LHS sa sadržajem RM rezultira vezivanjem varijabli koje su supstituirane na RHS prije izvršavanja pravila

OBLICI PRAVILA

Primjer

IF (auto ?x) AND (svjetla ?x slaba)
THEN (provjeri_baterije ?x)
RM: (auto V90)
(svjetla V90 slaba)

Rezultat paljenja pravila (provjeri_baterije V90)

PS koji dozvoljavaju varijable na obje strane su izražajniji ali manje efikasni – treba više vremena zbog podudaranja uzoraka

OBLICI PRAVILA

Primjer

AKO organizam = streptokoki ili organizam = gonorrhea
TADA indicirani_lijek = penicilin

AKO indicirani-lijek = penicilin i nepoznato(alergija_na = penicilin)
TADA pitaj(alergija_na = penicilin)

AKO indicirani_lijek = penicilin i nije(alergija_na = penicilin)
TADA prepisani_lijek = penicilin

Zadnja dva pravila moraju se ponavljati za svaki indicirani lijek ponaosob (na primjer, za aspirin, tetraciklin itd.)

OBLICI PRAVILA

Uvođenjem varijable **?lijek** na obje strane – dva pravila pokrivaju sve slučajeve:

AKO indicirani-lijek = **?lijek** i nepoznato(alergija_na = **?lijek**)
TADA pitaj(alergija_na = **?lijek**)

AKO indicirani_lijek = **?lijek** i nije(alergija_na = **?lijek**)
TADA prepisani_lijek = **?lijek**