

## Ostali formalizmi za predstavljanje znanja

### Sadržaj:

- Produkcioni sistemi
  - Produkciona pravila
  - Ekspertni sistemi
- Semantičke mreže i okviri

## Produkcioni sistemi

**Produkciona pravila** - razvijena zbog neefikasnosti rezolucije kao pravila izvodjenja

### Opsti oblik pravila:

IF situacija s THEN akcija a

IF uslov u THEN posledica p

If p THEN s TO DEGREE d

**Prednosti pravila:** modularnost, dopunjivost, jednostavnost, izmenjivost

### Arhitektura produkcionih sistema

- **Interfejs**
- **Baza znanja:** produkciona pravila (If-Then pravila)
- **Radna memorija:** činjenice
- **Mehanizam zaključivanja**
  - Lančanje unapred (forward chaining) data-driven  
Sve činjenice dostupne na početku. Pogodno za sintezu
  - Lančanje unazad (backward chaining) goal-driven  
Sva ciljna stanja dostupna na početku. Pogodno za dijagnostiku
  - U oba smera (bi-directional)

## Zaključivanje kod produkcionih sistema

Procesom pretrage upravlja strategija kontrole koja odlučuje koji se operator izvršava sledeći. Bitna osobina procesa pretrage je smer prema kome se zaključivanje nastavlja. U radu ekspertnog sistema izuzetno je važna strategija pretraživanja koja se koristi da se razvije traženo znanje. Razlikujemo dve osnovne vrste zaključivanja: unapred i unazad

### 1. Zaključivanje unapred (forward chaining) - početak → cilj;

Kreće se od početnog stanja ka ciljnom stanju (stanjima). Ova vrsta zaključivanja je vođena podacima (zavisno od vrednosti parametara stanja u datom čvoru definiše se put pretrage).

Lančanje unapred: polazi se od činjenica iz radne memorije, upoređuju se sa pravilima, okidaju se (primenjuju) pravila čiji je uslov zadovoljen (IF deo pravila), i zaključak (THEN deo pravila) se dodaje radnoj memoriji.

Produkcionni sistemi su oni koji rade na principu zaključivanja unapred (FORWARD CHAINING).

**Winston-ov algoritam:**

- Sve dok se problem ne reši ili nema pravila čiji se uslovi mogu zadovoljiti u trenutnoj situaciji radi:
  - Pronađi pravila čiji su uslovi zadovoljeni (ako ih ima više, primeniti postupak za razrešenje konflikta)
  - Izvršiti akcije koje su pridružene aktiviranim pravilima

**Algoritam zaključivanja:**

1. **Match**: Pronadji sva primenljiva pravila (If deo je zadovoljen)
2. **Conflict resolution**: Izaberi pravilo koje ce se izvršiti
  - Odbaci pravila koja su već ranije primenjena
  - Izaberi novo pravilo (*conflict-resolution strategy*)
3. **Act**: Dodaj novu činjenicu u WM

## 2. **Zaključivanje unazad** (backward chaining) - cilj → početak;

Kretanje se vrši od ciljnog stanja ka početnom stanju. Ovakvo zaključivanje je ciljno orijentisano (ne vodi računa kojim putem će se kretati, ali zna dokle mora da stigne).

Lančanje unazad: polazi se od ciljnih činjenica, upoređuju se sa THEN delom pravila, primenjuju se pravila čiji je zaključak zadovoljen, uslovi (činjenice iz IF dela) se dodaju radnoj memoriji kao novi ciljevi.

3. Kao kombinacija navedena dva načina pretraživanja, u nekim slučajevima se uspešno koristi **dvosmerno pretraživanje**: unapred od početnog stanja i unazad od ciljnog (ciljnih) stanja. Pretraga se završava kada se ova dva procesa pretrage "sretnu" na nekom delu grafa.

## Strategije za razrešavanje konflikata

- Primenuje se samo jedno pravilo, npr. prvo u bazi čiji je uslov ispunjen
  - No duplication: izaberi pravilo koje daje nove zaključke
  - Do one: izaberi prvo pravilo čiji je uslov ispunjen
  - Do the most specific: primeni najspecifičnije pravilo. Specifičniji je uslov koji sadrži više premisa.
  - Do the most recent: izaberi pravilo čiji uslov koristi najnovije činjenice
- Do all: primeni sva pravila, svi dobijeni zaključci se dodaju bazi činjenica
- Do in sequence: Primena pravila jedno za drugim tako da svaki zaključak može da se iskoristi za ispunjenje uslova novog ili sledećeg pravila

## Problemi sa produkcionim pravilima:

- Velike baze znanja mogu biti spore
- Mala količina informacija na osnovu kojih se donosi zaključak
- Pravila mogu biti redundantna

**Primer:**

Radna memorija = (zeleno, tesko-8kg)

Baza znanja:

- P1: IF zeleno, THEN povrce-voce
- P2: IF upakovano-u-male-kontejnere THEN delikates
- P3: IF rashladjeno OR povrce-voce, THEN kvarljivo
- P4: IF tesko-8kg AND jeftino AND NOT kvarljivo, THEN osnovna-namirnica
- P5: IF kvarljivo AND tesko-8kg THEN curka
- P6: IF tesko-8kg AND povrce-voce THEN lubenica

Za svaki ciklus, odredi:

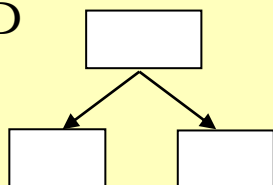
- Pravila koja su primenljiva
- Pravila koja treba de-aktivirati
- Izabrano pravilo, ako se izabira pravilo sa najnižim brojem
- Novo stanje radne memorije

## Grafički prikaz procesa zaključivanja

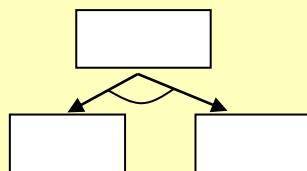
Za grafički prikaz procesa zaključivanja koriste se AND/OR stabla.

Grafička notacija:

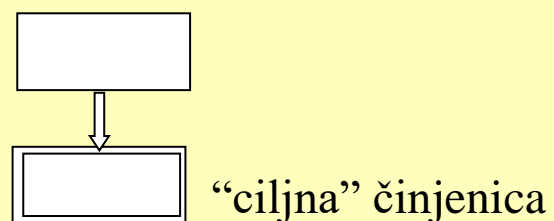
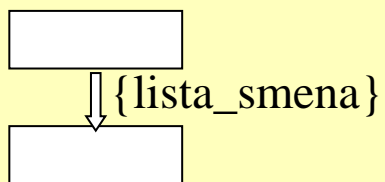
AND



OR



Primena pravila:





**Primer:** Nacrtati AND/OR stablo za zaključivanje lančanjem unapred

Data su pravila:

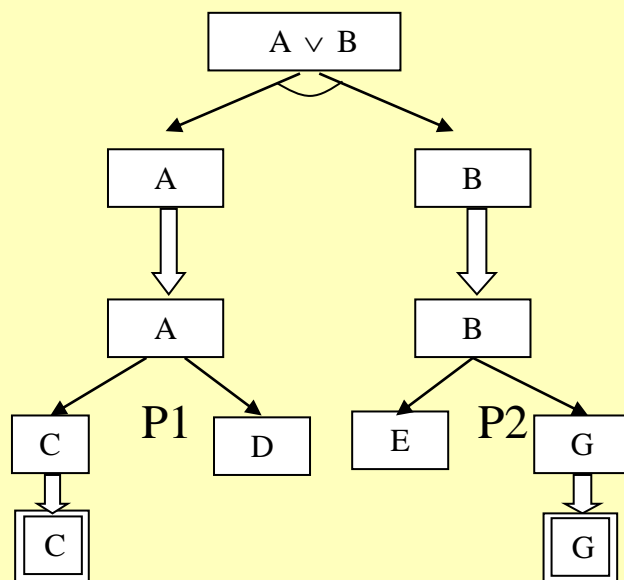
P1:  $A \Rightarrow C \wedge D$

P2:  $B \Rightarrow E \wedge G$

Činjenice:  $A \vee B$

Dokazati:  $C \vee G$

**Rešenje:**



**Primer:** AND/OR stablo za lančanje unazad:

Primer produkcionog sistema za dokazivanje nejednakosti

Predikat :

$V(x, y) \leftarrow x \text{ je veće od } y$

Skup pravila

P1:  $(V(x, 0) \wedge V(y, 0)) \Rightarrow V(\text{puta}(x, y), 0)$

P2:  $(V(x, 0) \wedge V(y, z)) \Rightarrow V(\text{plus}(x, y), z)$

P3:  $(V(x, w) \wedge V(y, z)) \Rightarrow V(\text{plus}(x, y), \text{plus}(w, z))$

P4:  $(V(x, 0) \wedge V(y, z)) \Rightarrow V(\text{puta}(x, y), \text{puta}(x, z))$

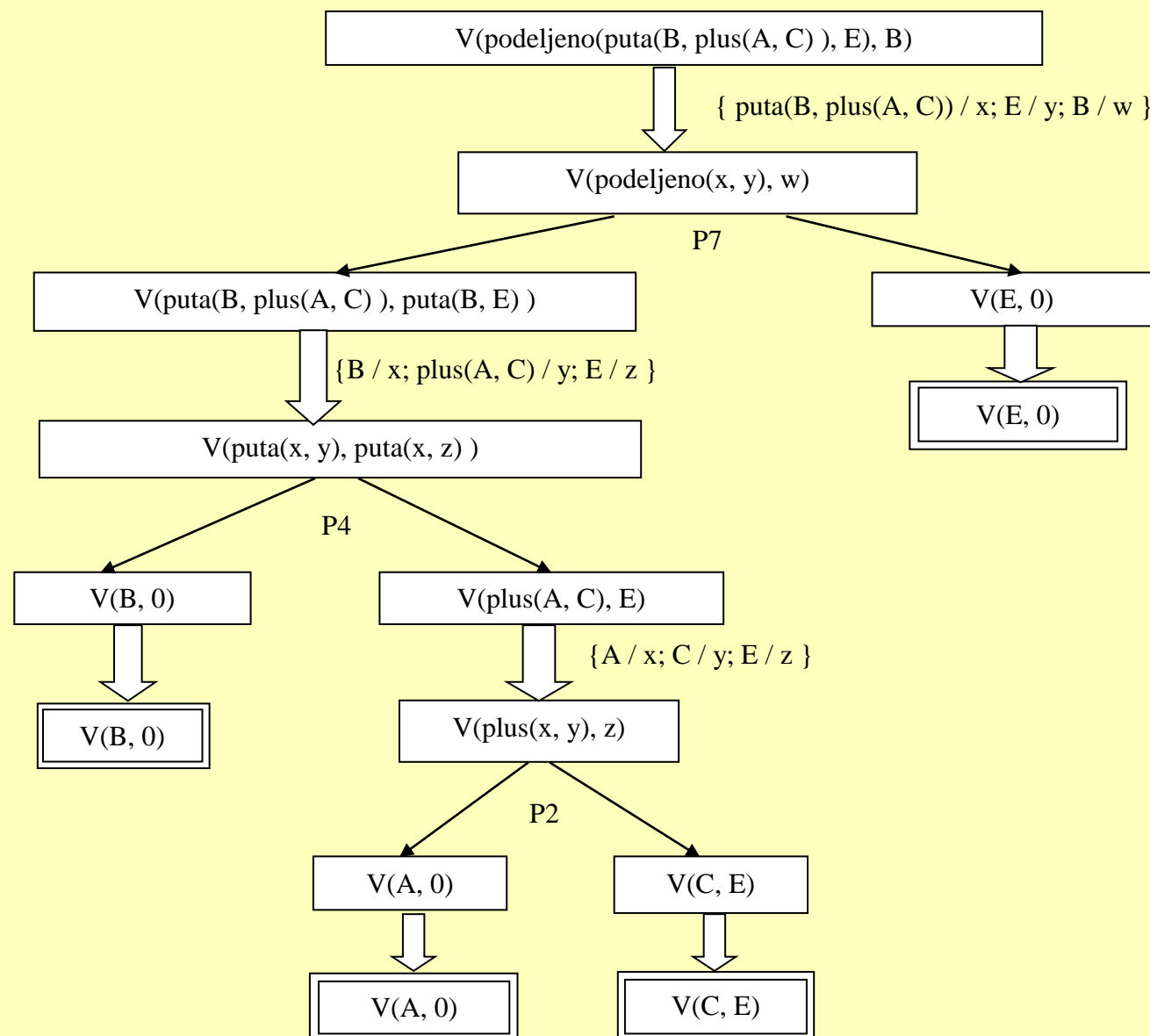
P5:  $(V(1, w) \wedge V(x, 0)) \Rightarrow V(x, \text{puta}(x, w))$

P6:  $V(x, \text{plus}(\text{puta}(w, z), \text{puta}(y, z))) \Rightarrow V(x, \text{puta}(\text{plus}(w, y), z))$

P7:  $(V(x, \text{puta}(w, y)) \wedge V(y, 0)) \Rightarrow V(\text{podeljeno}(x, y), w)$

Činjenice:  $E > 0, B > 0, A > 0, C > E, C > 0$

Dokazati:  $(B(A + C) / E) > B$



**Primer:** Primer sistema za lančanje unapred.

Identifikacija životinja po opisu.

P1: Ako mladunci\_sisaju Onda je\_sisar

P2: Ako ima\_perje Onda je\_ptica

P3: Ako ima\_krzno ili je\_sisar Onda živi\_u\_šumi

P4: Ako je\_ptica i ne\_leti i ne\_živi\_u\_šumi Onda je\_pingvin

P5: Ako živi\_u\_šumi i veoma\_je\_teška Onda je\_medved

P6: Ako veoma\_je\_teška i je\_sisar Onda je\_kit

**Interpretator** ( mehanizam za zaključivanje ) radi u 4 koraka:

- 1) interpretator nađe sva pravila kod kojih je uslov zadovoljen
- 2) ako ima više takvih pravila ignorišu se ona koja daju poznate osobine
- 3) izvršiti akciju pravila sa najnižim rednim brojem. Ako takvog pravila nema prestaje se sa obradom.
- 4) Vрати se na 1).

Činjenice: {mladunci\_sisaju, veoma\_je\_teška, ima\_krzno}

1) P1, P3  $\rightarrow$  {mladunci\_sisaju, veoma\_je\_teška, ima\_krzno, je\_sisar} // bira se P1

2) P1, P3, P6  $\rightarrow$  {mladunci\_sisaju, veoma\_je\_teška, ima\_krzno, je\_sisar, živi\_u\_šumi}

3) P1, P3, P5, P6  $\rightarrow$  {mladunci\_sisaju, veoma\_je\_teška, ima\_krzno, je\_sisar, živi\_u\_šumi, je\_medved}

4) P1, P3, P5, P6  $\rightarrow$  {mladunci\_sisaju, veoma\_je\_teška, ima\_krzno, je\_sisar, živi\_u\_šumi, je\_medved, je\_kit}

$\Rightarrow$  Skup pravila je nepotpun, jer se dobija *je\_medved* i *je\_kit*, pa se onda skup pravila proširuje.

## **Ekspertni sistemi**

Sistemi bazirani na znanju koji resavaju ili pomazu pri resavanju problema u usko specijalizovanim domenima, na način kako to rade eksperti.

Ekspertni sistem je računarska aplikacija koja teži da oponaša ponašanje (rad) čoveka eksperta - stručnjaka koji poseduje izuzetno specijalističko znanje (u određenoj oblasti). Primenujući tehnike veštačke inteligencije, ekspertni sistemi usvajaju osnovno znanje koje čoveku omogućava da se ponaša kao ekspert prilikom rešavanja složenih problema. Ekspertni sistem simulira proces ljudskog mišljenja primenom određenog znanja i zaključivanja.

### **Karakteristike eksperata:**

- Poseduju znanje organizovano na efikasan način
- Postavljaju i testiraju hipoteze
- Mogu da objasne zasto i kako
- Mogu da efikasno rade čak i sa nepouzdanim podacima

### **Karakteristike ekspertnih sistema:**

- Emuliraju ponašanje eksperata
- Resavaju kompleksne probleme
- Mogu da reformulišu problem u toku rada
- Mogu da objasne postupak rešavanja

**Klase primene ekspertnih sistema:**

- Interpretacija podataka
- Predviđanje
- Projektovanje
- Nadgledanje procesa
- Planiranje

**Područja primene:**

medicina, industrija, saobraćaj, pravo, ekonomija ...

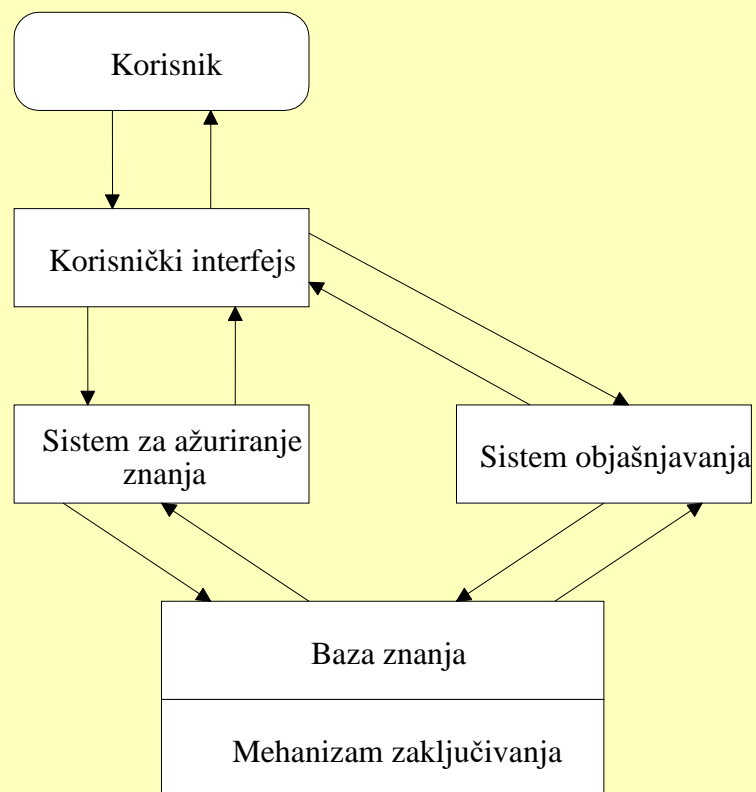
**Rezultati uvođenja ekspertnih sistema:**

- Ubrzanje rada
- Finansijski efekat
- Novi proizvodi i usluge
- Pобољшanje kvaliteta odlučivanja

**Uspesni ekspertni sistemi:**

- MYCIN (Shortliffe 1964): dijagnosticiranje i terapija infektivnih bolesti na osnovu simptoma
- DENDRAL (Feigenbaum & Buchanan 1965): određivanje hemijske strukture na osnovu spektralnih podataka
- INTERNIST (Pople 1968) interna medicina
- R1 – XCON (McDermott 1980) konfiguracija DEC računara

## Arhitektura ES



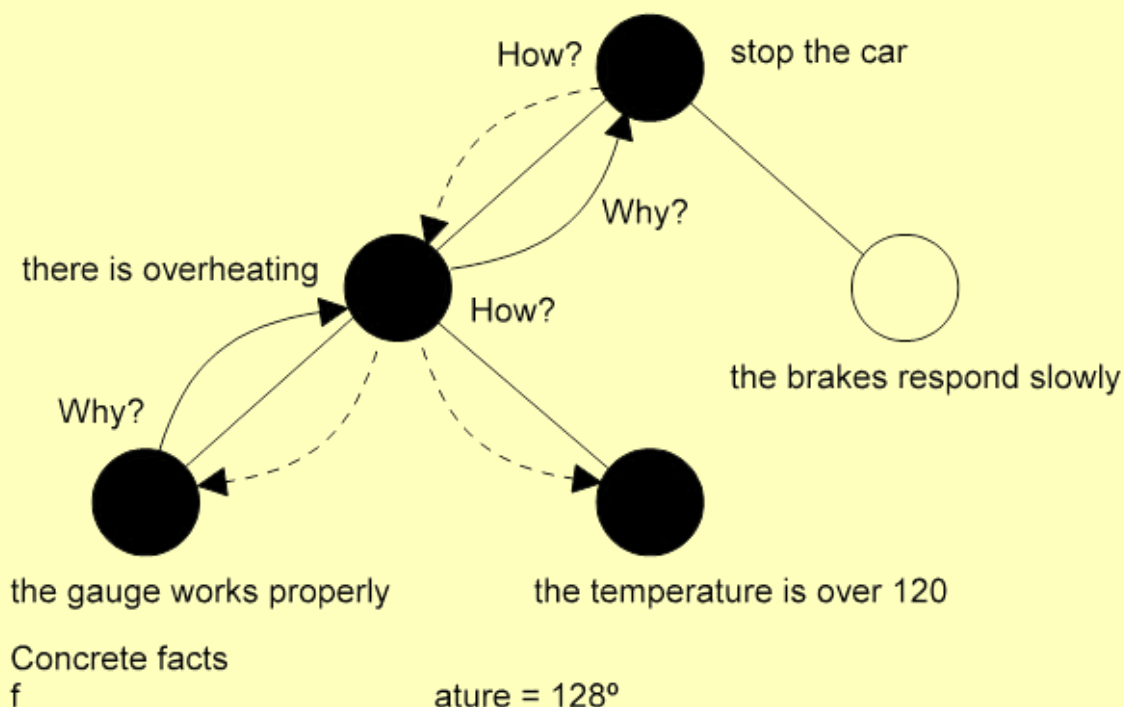
Slika: Arhitektura ekspertnog sistema



**Elementi ekspertnog sistema su:**

1. Korisnik - osoba koja koristi ili održava ekspertni sistem.
2. Korisnički interfejs - održava se dvosmerna transformacija informacija na relaciji korisnik - sistem, uz stalnu težnju ka korišćenju govornog jezika.
3. Sistem za skladištenje i generisanje znanja
  - 3.1. Baza znanja - skladište primitiva znanja koja su dostupna sistemu. Znanje se najčešće skladišti u obliku činjenica i pravila, ali postoje i drugi načini.
  - 3.2. Mehanizam (mašina) za zaključivanje - softverski sistem koji pronalazi znanje i izvodi novo iz postojećeg znanja. Ovaj element određuje stepen fleksibilnosti ekspertnog sistema, odnosno mogućnost da se izvede novo znanje iz postojećeg. Na primer, iz dve primitive “Životinje udišu kiseonik” i “Psi su životinje” proizilazi zaključak “Psi udišu kiseonik”. Znanje koje je potrebno za donošenje odluke može biti postojeće ili izvedeno. Idealan ekspertni sistem (njegov mehanizam zaključivanja) je onaj koji nikada neće biti potrebno modifikovati. Sama eksploatacija ekspertnog sistema ne dovodi do promene mehanizma zaključivanja, već samo do njegovog korišćenja.

4. Sistem za ažuriranje znanja - zajedno sa napretkom nauke i tehnike, neprestano napreduje i znanje iz tih oblasti. Ove promene povlače za sobom i promene znanja u bazi ekspertnog sistema. Znanje u bazi znanja ekspertnog sistema ručno može da ažurira (menja, dopunjava i briše) inženjer baze znanja ili ekspert lično, a što se vrši preko ove komponente. Treći način ažuriranja znanja je mašinsko učenje - situacija kada ekspertni sistem uči na prethodnog iskustva, pa se sistem samo-ažurira.
5. Sistem za objašnjavanje - preko ove komponente ekspertni sistem treba da, kao i svaki stručnjak, pruži informacije o postupku koji ga je doveo do određenog zaključka. Ovo je bitna osobina koja nedostaje tradicionalnim kompjuterskim sistemima. Sam način objašnjavanja treba da bude različit za svakog korisnika pojedinačno, zavisno od nivoa njegovog znanja. (Slika: HOW and WHY objašnjenje rada ES za monitoring automobila)



### **Preporuke za izgradnju ekspertnih sistema:**

- Izaberi zadatak koji nije ni pretezak ni lak
- Zadatak mora biti jasno definisan
- Neophodno je dobro upoznati problem pre komunikacije sa ekspertom
- Komunikacija sa ekspertom je ključni factor
- Protokoli resavanja problema su neophodni
- Pomocni alati su jako vazni
- Realizacija mora da pocne sa prototipom

### **Alati za izgradnju ekspertnih sistema**

- Manji sistemi (do 400 pravila): Goldworks, Insight, Personal Consultant
- Veci sistemi (do hiljadu pravila) Expert, KES, OPS5, TIMM
- Veliki sistemi: ART, KEE, Loops, Nexpert Object

### **Evolutivni razvoj ekspertnih sistema:**

- Demonstracioni prototip

Veoma ograničena verzija sistema, obavlja samo jednu funkciju. Baza znanja od 50 do 100 pravila. Služi za ispitivanje ostvarljivosti i funkcionalnosti sistema, za upoznavanje sa domenom

- Istraživački prototip

Veća baza znanja (do 200 pravila). Može da obavi više funkcija

- Eksperimentalni prototip

Potpuno razvijena baza znanja. Izvršava sve funkcije, namenjen tetiranju od većeg broja korisnika

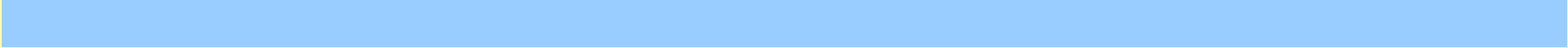
- Industrijski system

Od eksperimentalnog prototipa se razlikuje po bržem i efikasnijem radu

**Prednosti** ekspertnog sistema u odnosu na rešenje problema klasičnim programiranjem su:

- mogućnost objašnjavanja postupka rešavanja,
- ugradjena fleksibilnost u program,
- mogućnost rada sa nepotpunim, netačnim, nepouzdanim podacima,
- mogućnost popravljavanja i dopune baze znanja,
- mogućnost nalaženja rešenja za složene i formalno nejasne probleme.

**Osnovni nedostatak** ekspertnog sistema je kombinatorna eksplozija. U potrazi za rešenjem, broj svih mogućnosti formiranih od nekoliko osnovnih znanja (primitiva) je izuzetno veliki, tako da nadmašuje memoriju računara i dozvoljeno vreme za rešavanje datog problema. Prevazilazi se korišćenjem “kompajliranog” (sažetog) znanja eksperta koje se stvaralo duži niz godina (radno iskustvo eksperta) (ne radi se sa primitivama - osnovnim znanjem).



## Semantičke mreže

Formalizam za predstavljanje znanja, nastao u cilju razumevanja načina dugoročnog pamćenja kod ljudi.

Semantičke mreže opisuju kategorije objekata (čvorovi u mreži) i relacije medju njima (potezi). Glavni oblik zaključivanja je nasledjivanje.

Svaki čvor u mreži ima ime, koje ne mora da bude jedinstveno.

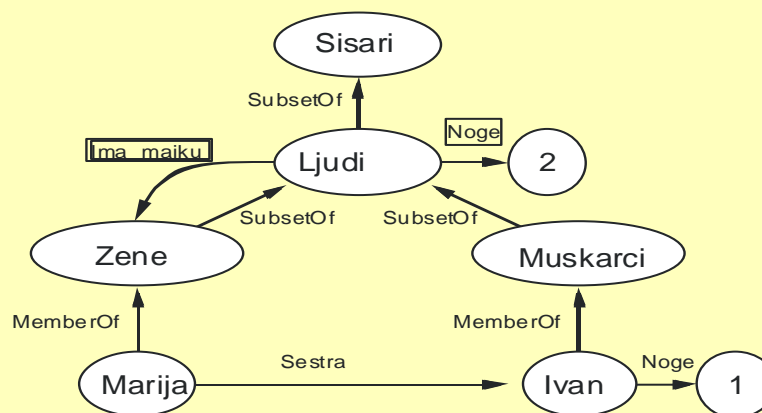
Opisuju se prototipovi, a moguće je opisati i izuzetke (nemonotona logika)

### Tipovi veza:

- SubsetOf (a-kind-of)
- MemberOf (is-a)
- Relacija izmedju dva objekta (strelica sa imenom relacije)
- Relacija izmedju svakog elementa klase A i objekta B (ime relacije uokvireno)
- Relacija izmedju svakog elementa klase A i nekog elementa klase B (ime relacije dvostruko uokvireno)

## Implementacija

- Stvarne vrednosti
- Default vrednosti
- Procedure (if needed, if added)



Slika: Primer semantičke mreže

## Prevodjenje u logiku predikata:

$Ljudi \subset Sisari$

$Marija \in Zene$

$Noge(Ivan, 1)$

$\forall x \ x \in Ljudi \Rightarrow Noge(x, 2)$

$\forall x \ \exists y \ x \in Ljudi \Rightarrow y \in Zene \wedge Ima\_majku(x, y)$

## Procedure

Procedure se mogu dodati u mrežu, da bi se izračunale neke vrednosti (procedural attachment)

Procedure za:

- dodavanje i brisanje čvora
- dodavanje i brisanje potega
- pretraživanje
- if-added
- if-needed

## Prednosti semantičkih mreža:

- Pogodne za hijerarhijski uređene podatke
- Podrazavaju način organizacije memorije kod ljudi
- Pogodne za unarne i binarne relacije
- Nasleđivanje

Za semantičke mreže važi pravilo nasleđivanja osobina koje govori da svaka osobina koja važi za klasu elemenata takođe treba i da važi za svaku instancu klase.



## Nedostaci semantičkih mreža

- Teškoće pri predstavljanju relacija višeg stepena
- Disjunkcija i negacija se tesko predstavljaju
- Teško se izvodi kvantifikacija

## Zaključivanje:

Traženje presekom - aktiviranjem čvorova mreže počev od oba čvora za koja se traži da li su povezani (dva pojma); ako presek postoji, postoji veza između pojmova.

Zaključivanje pomoću semantičkih mreža je uglavnom unapred usmereno (STRAIGHT FORWARD) jer se asocijacije lako formiraju praćenjem veza u sistemu. Na žalost, moguće je formiranje zaključaka koji zanemaruju pojedine izuzetne situacije, dok kod predikatske logike neki rezultati procesa zaključivanja mogu biti nevažni, ali su uvek validni (ispravni).

Zaključivanje na osnovu semantičkih mreža se može vršiti metodom nasleđivanja putem uklapanja - gradi se deo mreže sa čvorovima koji imaju poznate i nepoznate vrednosti. Procedura uklapanja traži onu mrežu koja će savršeno odgovarati konstruisanom fragmentu. Često je potrebno da procedura uklapanja nasledi nove veze od postojećih veza kada konstruiše poseban fragment.

## Okviri

Složeni formalizam za predstavljanje znanja koji opisuje veće i složenije jedinice znanja.

Ljudi često na novu situaciju reaguju primenom očekivanja koja su zasnovana na prošlom iskustvu.

Okvir je struktura za organizovanje znanja, posebno podrazumevanog znanja. Svaki okvir predstavlja jednu klasu elemenata na isti način na koji se koristi čvor klase da predstavi takve elemente u semantičkoj mreži.

Okvir opisuje entitet preko njegovih *osobina*. Svaka osobina ima vrednost koja se smešta u *slot* okvira. Vrednosti okvira mogu ali ne moraju biti specificirane. Za neke slotove se mogu vezati procedure koje treba aktivirati u određenim situacijama.

Okviri su uređeni u hijerarhije preko vrednosti nekog slot-a. Vrednosti slotova se mogu nasleđivati ili se mogu dodeliti nove vrednosti.

Postoji i **proceduralno znanje**: za svaki slot se vežu određene procedure koje se izvršavaju u određenim situacijama. Okviri se uređuju u hijerarhije na osnovu veze is-a.

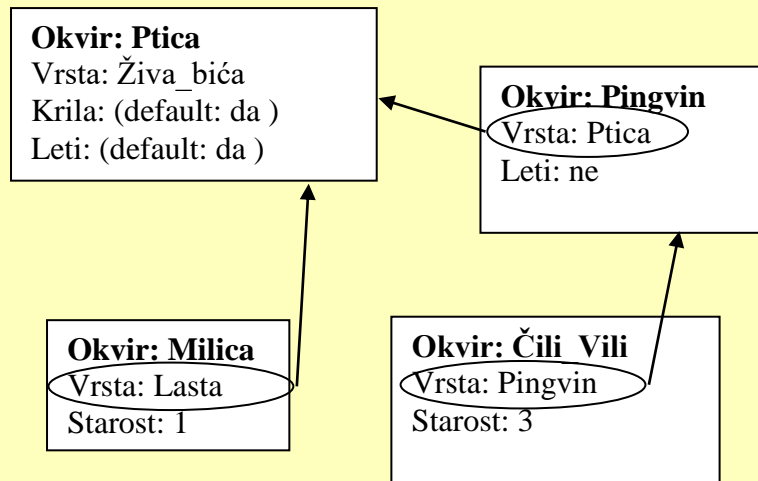
Moguće je *kombinovati* **deklarativno** (vrednosti slotova) i **proceduralno** znanje (vezane procedure).

Okvir:	Automobil
Vrsta:	Kopneno vozilo
Telo:	Čelik
Prozori:	Staklo
Gorivo:	(Super, Dizel, Gas)
Broj sedišta:	(1 - 9)

Slika: Primer okvira

### Vrste procedura koje se koriste u okvirima:

1. IF-ADDED definiše postupak koji se obavlja kod dodeljivanja vrednosti datom slotu. Procedura mora uspešno da se izvrši da bi se upisala vrednost.
2. IF-NEEDED definiše postupak prilikom pretraživanja vrednosti slotu. Mora uspešno da se izvrši da bi se pročitala vrednost.
3. IF-UPDATED definiše postupak prilikom editovanja vrednosti slotu.

**Primer:** Hijerarhija okvira (nasleđivanje preko vrednosti slota *Vrsta*):**Vrste nasleđivanja:**

**I nasleđivanje** - čitanje iz okvira se vrši tako što se traži element vrednost u prvom okviru, a zatim se traži isti taj element u okviru roditelju, itd.

**N nasleđivanje** - čitanje iz okvira se vrši tako što se traži element vrednost po celoj hijerarhiji okvira, zatim se vraća na početni okvir, pa se traži default po celoj hijerarhiji i ako nema vraćamo se na prvi okvir i traži IF-NEEDED procedura, itd.

**Z nasleđivanje** - čitanje iz okvira pri čemu se traži vrednost default i IF-NEDDED u prvom okviru, a onda se traži po hijerarhiji.

## Poređenje formalizama za predstavljanje znanja

### 1. Ekspresivnost

Logika i produkcionni sistemi mogu da se koriste nezavisno

Semantičke mreže se obično koriste u hibridnim sistemima

### 2. Semantika

Logika ima jasnu semantiku. Produkcionni sistemi i mreže su ponekad nejasni

### 3. Izvršavanje

Produkcionni sistemi su najjednostavniji

Semantičke mreže su komplikovane ako se koriste i procedure

Logika zahteva komplikovane procedure za zaključivanje