UNIVERZITET U NOVOM SADU

FAKULTET TEHNIČKIH NAUKA

Nastavni predmet:

INTEGRISANI CAPP SISTEMI I TEHNOLOŠKA BAZA PODATAKA

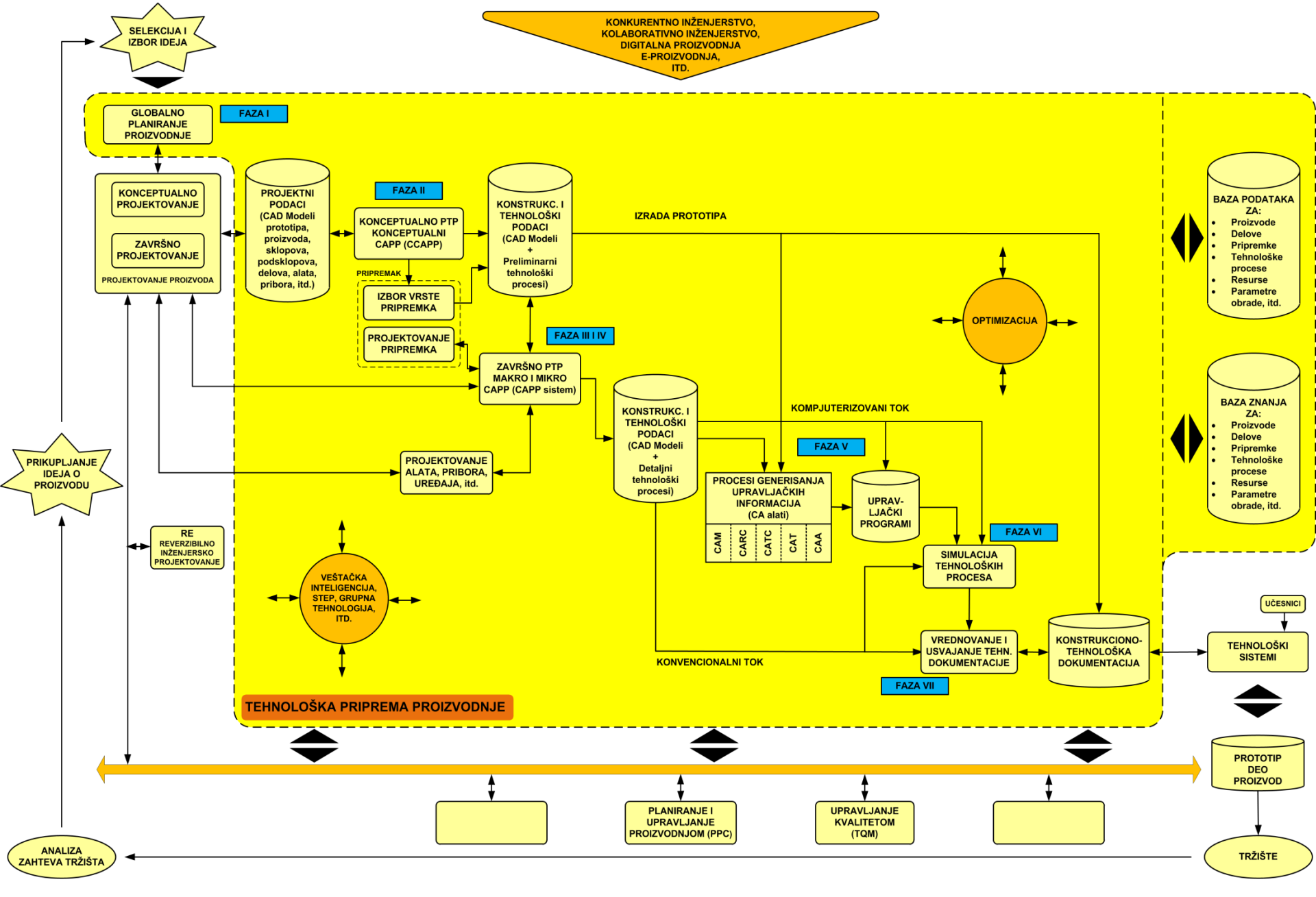
Predavanja br. 11:

Savremene metode i tehnike razvoja i integracije CAPP Sistema

Tema: Metode veštačke inteligencije i agent bazirane metode

Prof. dr Dejan Lukić

Opšti model tehnološke pripreme proizvodnje-faze III i IV I V



U razvoju CAPP sistema i njihovoj integraciji sa drugim funkcijama i aktivnostima proizvodnog sistema i globalnog poslovnog okruženja primenjuju se brojne metode i tehnike, koje se mogu koristiti zasebno ili integralno. Na osnovu analize brojnih literaturnih informacija, izdvojen je set osnovnih savremenih metoda i tehnika koje se koriste za razvoj CAPP sistema:

- ***Metode zasnovane na tipskim oblicima,***
- ***Metode veštačke inteligencije***
  - ***Ekspertni sistemi,***
  - ***Neuronske mreže,***
  - ***Genetski algoritmi,***
  - ***Fuzzy teorija i fuzzy logika,***
- ***Agent-bazirane metode,***
- ***Internet-bazirane metode,***
- ***Metode bazirane na STEP standardu, i dr.***

Primena metoda veštačke inteligencije u razvoju CAPP sistema

Veštačka inteligencija (VI) kao naučna disciplina se odnosi na razvoj inteligentnih računarskih sistema, koji imaju karakteristike povezane sa inteligentnim ponašanjem čoveka, kao što su razumevanje, učenje, razmišljanje, rešavanje problema, i dr.

U razvoju i primeni CAPP sistema i njihovoj integraciji sa drugim aktivnostima proizvodnog sistema značajnu primenu imaju metode veštačke inteligencije, koje se u osnovi odnose na koncept inženjerstva znanja. Pod konceptom inženjerstva znanja podrazumevamo sledeće zadatke: *akviziciju znanja, predstavljanje znanja, izvođenje zaključaka i transfer znanja do korisnika.*

Znanje je ključni činilac u odlučivanju, upravljanju i uopšte rešavanju problema. Akvizicija znanja je jedna od formi mašinskog učenja i predstavlja sposobnost sistema VI za sticanje i razvoj novog znanja ili unapređenje postojećeg znanja posredstvom ulaznih podataka i informacija koje zadaje i unosi čovek.

Predstavljanje znanja se vrši preko *činjenica, pravila i hipoteza*. Znanje, odnosno činjenice, pravila i hipoteze se mogu odnositi na *objekte, događaje, relacije i procedure*, odnosno *postupke*. Tehnike koje se najviše koriste za predstavljanje baze znanja, odnosno logike za donošenje odluka u okviru pojedinih aktivnosti CAPP sistemima su: *matematička logika (račun iskaza i račun predikta), produkcioni sistemi, semantičke mreže, frejmovi i objektni način predstavljanja znanja.*

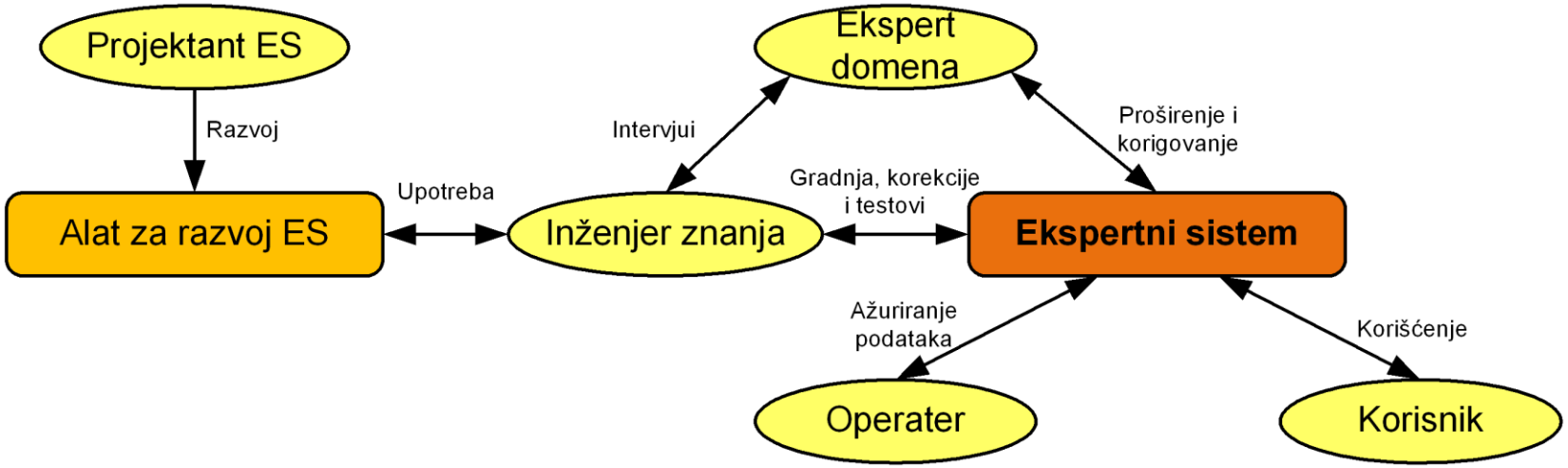
Izvođenje zaključaka se najčešće vrši na bazi pravila koja sadrže istinitu vrednost i odgovarajuću logičku zavisnost sa zaključkom. U teoriji odlučivanja primenjuju se tri osnovna tipa zaključivanja deduktivni, induktivno i na bazi analogije. Kada se govori o zaključivanju veoma često se misli na inteligentno zaključivanje i optimizaciju rešenja problema (multikriterijumski, nelinearni, diskretni i fuzzy).

Ekspertni sistemi

ES je inteligentni računarski program koji se koristi znanjima i procedurama zaključivanja radi rešavanja problema koji su dovoljno teški da njihovo rešenje zahteva stručnu ekspertizu.

ES podrazumeva uspostavljanje unutar računara deo veštine nekog eksperta koji bazira na znanju i u takvom je obliku da sistem (računar) može da ponudi inteligentan savet ili da preuzme inteligentnu odluku u funkciji koja je u postupku.

Razvoj ekspertnih sistema je baziran na znanju eksperata ili stručnjaka iz određene oblasti, koje treba prikupiti i predstavi u pogodnom obliku za računarsku implementaciju, što je najčešće zadatak koji realizuju inženjeri znanja. Zatim to znanje treba da se implementira kroz odgovarajući programski – ekspertni sistem.

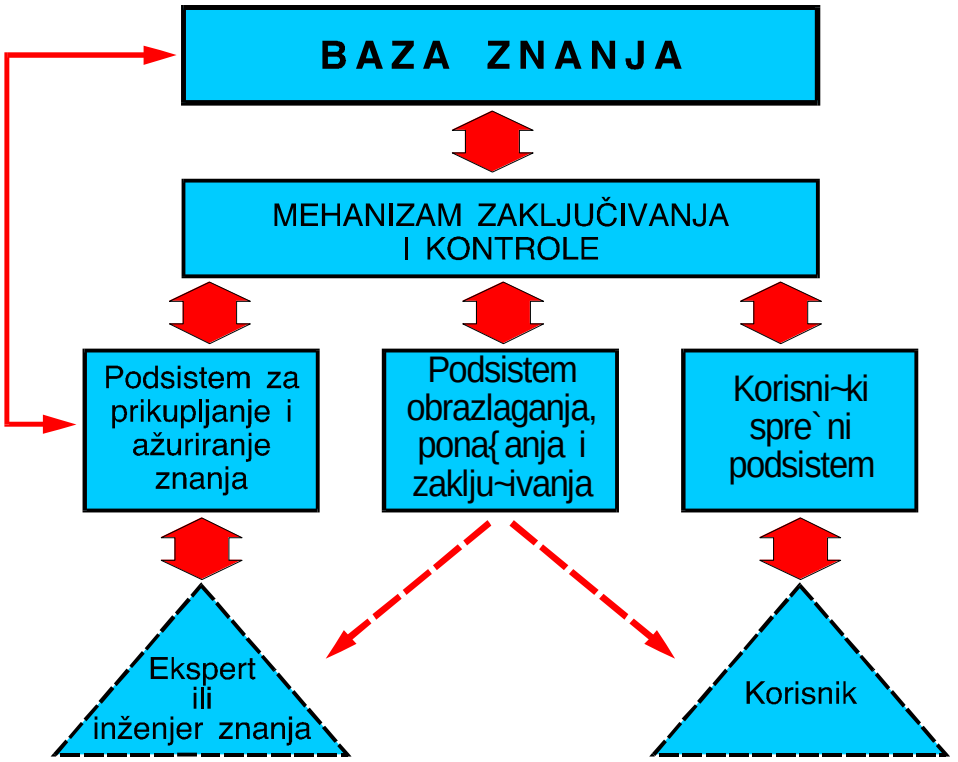


Osnovna struktura i učesnici u razvoju ekspertnih sistema

Arhitekturaktura ekspertnih sistema

Tri osnovne komponente koje sadrže svi ES su

- *Baze znanja (knowledge base),*
- *Mehanizmi zaključivanja (inference machine), i*
- *Korisnički interfejs.*



Baza znanja iz određene oblasti sastoji se od činjenica i heuristike, relacija između njih i metoda za rešavanje problema.

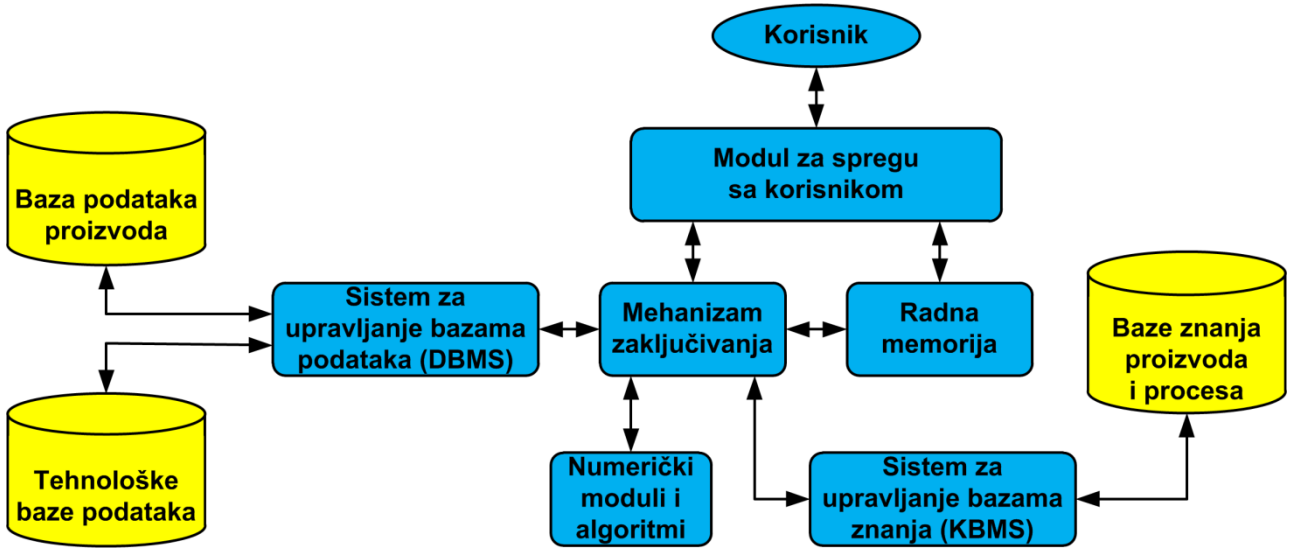
Mehanizam zaključivanja (inference machine) je deo ES koji uključuje strategije zaključivanja i kontrole za manipulisanje znanjem radi dolaženja do novih informacija i znanja, uz istovremeno rešavanje konfliktnih situacija.

Korisnički interfejs prihvata informacije koje zadaje korisnik i prevodi ih u oblik razumljiv za sistem, kao i obrnuto.

Podsistem za obrazlaganje ponašanja i zaključaka vrši identifikovanje koraka u procesu rezonovanja i proveru njihove ispravnosti. (U postojećim ES ova komponenta veoma često služi za izlistavanje pravila i proveru njihove sposobnosti). Podsistem za prikupljanje i ažuriranje omogućuje uvećanje i izmene u bazi znanja ES.

Opšta struktura ekspertnog CAPP sistema

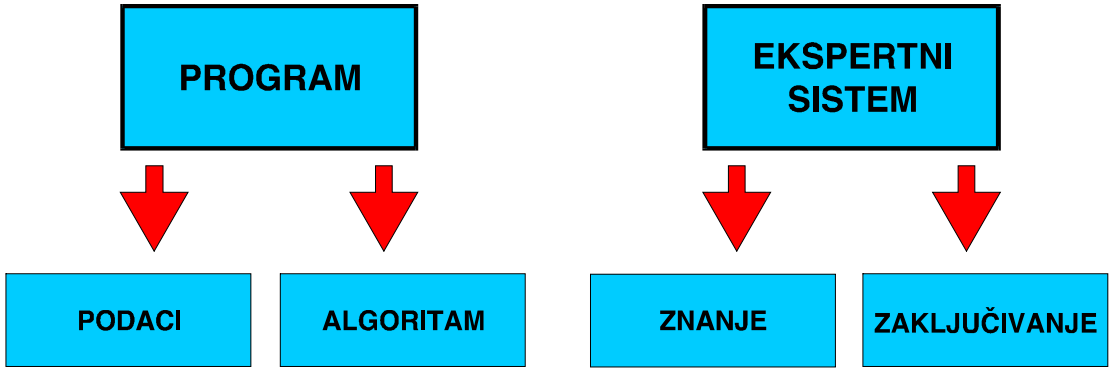
Na slici je prikazan model opšte strukture ekspertnog CAPP sistema, koji pored navedenih elemenata obavezno poseduje integrisane baze podataka i znanja ili odgovarajuće baze podataka i znanja za proizvode i pojedine elemente projektovanja tehnoloških procesa, najčešće proizvodne resurse.

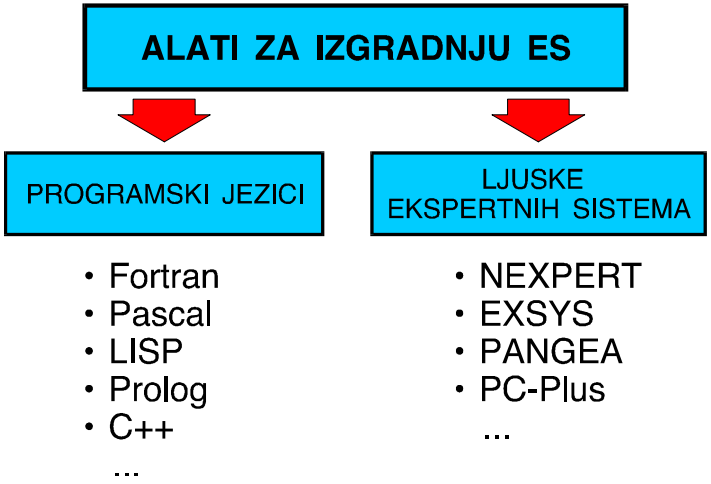


Znanje potrebno za rešavanje zadataka projektovanja tehnoloških procesa je kontekstno zavisno i kontekstno nezavisno, predstavlja se objektima (deklarativno znanje) i produkcionim pravilima (proceduralno znanje) i kao takvo je smešteno u odgovarajuće baze znanja. Ova baza znanja se sastoji od različitih vrsta znanja, od znanja za tehnološko prepoznavanje, preko znanja za izbor pripremaka, mašina, alata, pribora, itd.

U okviru posmatranog modela ekspertnog CAPP sistema, mehanizam zaključivanja vrši pretraživanje baza znanja i baza podataka, i upravlja čitavim tokom projektovanja tehnoloških procesa.

Način razvoja ekspertnih sistema





PROGRAM=PODACI+ALGORITAM

EKSPERTNI SISTEM=ZNANJE+ZAKLJUČIVANJE

ZNANJE=ČINJENICE+MIŠLJENJE+HEURISTIKA

OBRADA PODATAKA	INŽENJERING ZNANJA
- PREZENTOVANJE I KORIŠĆENJE PODATAKA	- PREZENTOVANJE I KORIŠĆENJE ZNANJA
- ALGORITMI	- HEURISTIKA
- PONAVLJANJE PROCESA	- MEHANIZAM ZAKLJUČIVANJA
- EFEKTIVNO MANIPULISANJE SA VELIKIM BAZAMA PODATAKA	- EFEKTIVNO MANIPULISANJE SA VELIKIM BAZAMA ZNANJA

Prednost ES iz ovog koncepta u odnosu na konvencionalne programe je u tome što u slučaju novog znanja nije potrebno programiranje, već samo proširenje baza znanja, koja najčešće nije sastavni deo programa.

Faze razvoja ekspertnih sistema

Razvoj ES podrazumeva izgradnju baze znanja – sticanje ili akviziciju i unošenje znanja u sistem.

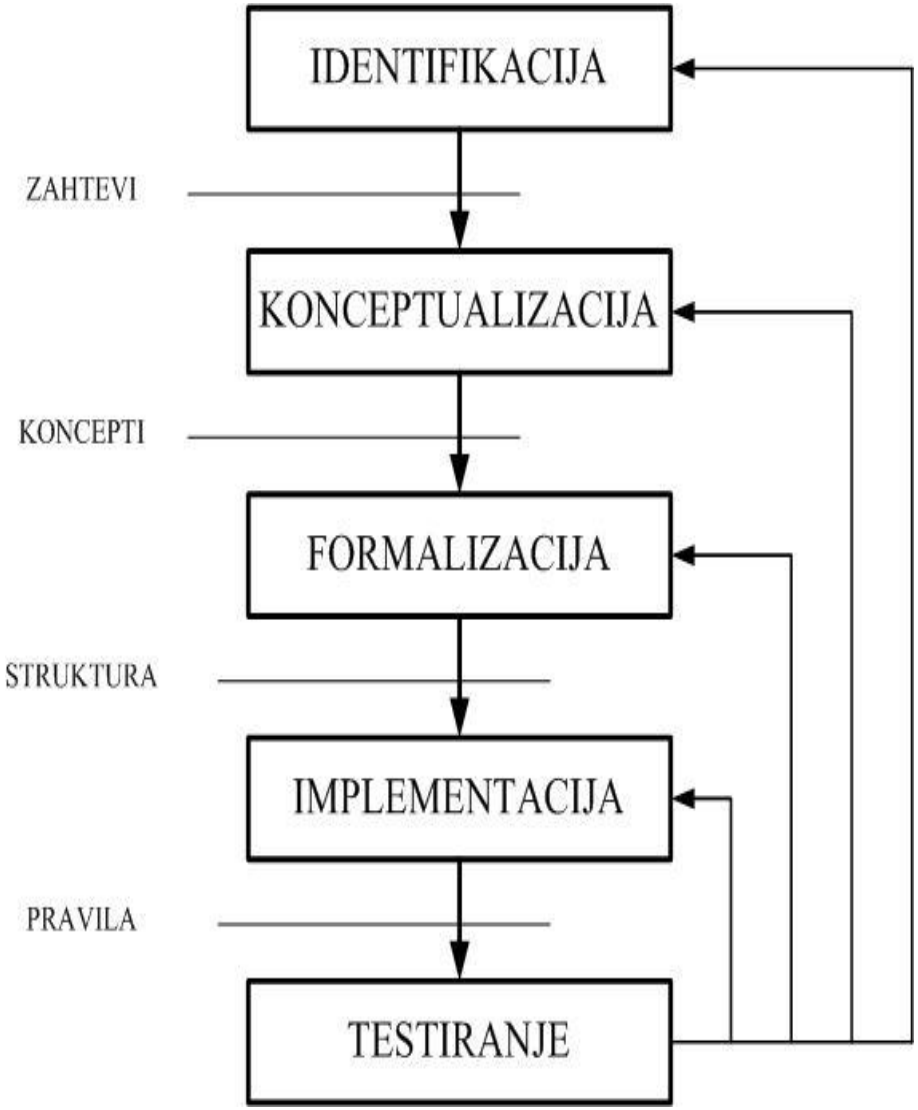
Faza identifikacije - identifikuje se zadatak, izučava postojeće znanje.

Faza konceptualizacije – inženjer znanja postavlja koncept pristupa, kojima se definišu činjenice i hipoteze na kojima je zasnovan zadatak.

Faza formalizacije – znanje se dovodi u oblik pogodan za predstavljanje (u programu i ekspertnoj ljusci).

Faza implementacije – formirane strukture se prenose u računar i njen produkt je baza znanja.

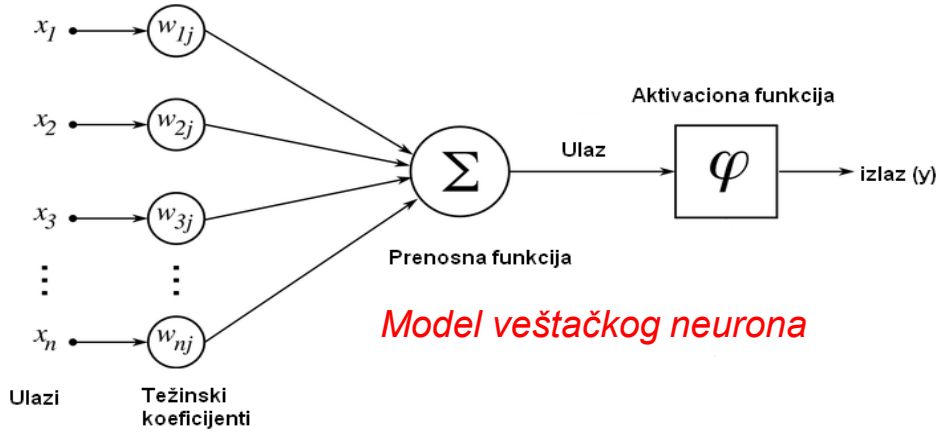
Faza testiranja – ekspert uočava semantičke nepravilnosti a inženjer znanja nepravilnosti u realizaciji procedura



Napomena: pogledati odgovarajuću vežbu

Neuronske mreže

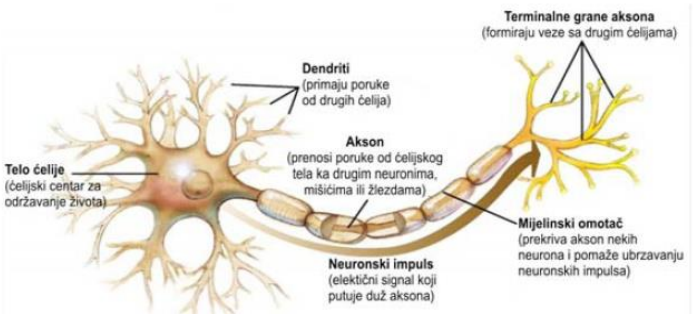
Neuronske mreže predstavljaju tipičan primer interdisciplinarne oblasti, koja je nastala kao rezultat spoja nekoliko različitih naučnih prilaza obrade signala, fizike, tehnike, neurobiologije i dr. Kada se govori o neuronskim mrežama prevashodno se misli na veštačke neuronske mreže (Artifical Neural Networks). Veštačke neuronske mreže predstavljaju skup jednostavnih procesirajućih elemenata – neurona, međusobno povezanih u paralelnu distribuiranu strukturu, koja simulira funkciju bioloških nervnih sistema, odnosno vrši obradu informacija i učenje.



Model veštačkog neurona

Gde su:

- $X_{i...n}$ – *ulazni podaci*
- $W_{j...n}$ – *težinski koeficijenti*
- *φ – aktivaciona funkcija*
- *v – izlazni podatak*



Matematički model neurona se sastoji iz dve jednačine:

$$ulaz = \sum_{i=1}^{n} w_j x_i \qquad izlaz(y) = f(ulaz)$$

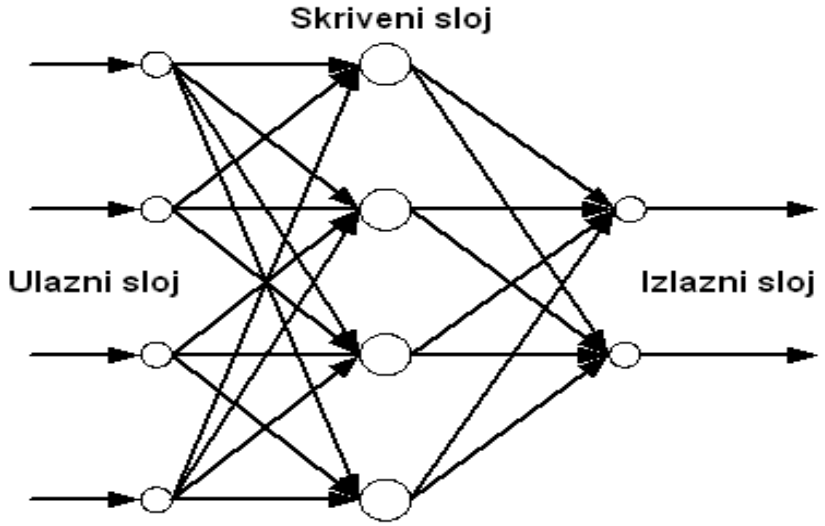
Kod veštačkog neurona sinapsa se karakteriše efikasnošću, koja se naziva težinski koeficijent W_j (sinaptička težina). Izlaz neurona se formira na sledeći način: signali na dendritima se pomnože odgovarajućim težinskim koeficijentima, proizvodi se saberu u prenosnoj funkciji *Σ* i ako prelaze veličinu praga, na dobijenu vrednost se primeni aktivaciona funkcija neurona *φ*.

Model neuronske mreže

Neuronska mreža je sistem koji se sastoji od više jednostavnih procesora (neurona), povezanih međusobnim vezama koje sadrže težinske koeficijente.
Model neuronske mreže čine:

- *Arhitektura ili topologija mreže,*
- *Prenosna funkcija neurona,*
- *Zakoni učenja, i*
- *Realizacija mreže.*

Arhitekturu, odnosno topologiju neuronske mreže predstavlja specifično uređenje i povezivanje neurona u obliku mreže. Neuronske mreže se međusobno razlikuju po broju neuronskih slojeva. Prvi sloj se naziva ulazni, a poslednji sloj je izlazni, dok se ostali slojevi između njih nazivaju skriveni slojevi. Obično svaki skriveni sloj prima ulaze od predhodnog sloja, a svoje izlaze šalje narednom sloju.

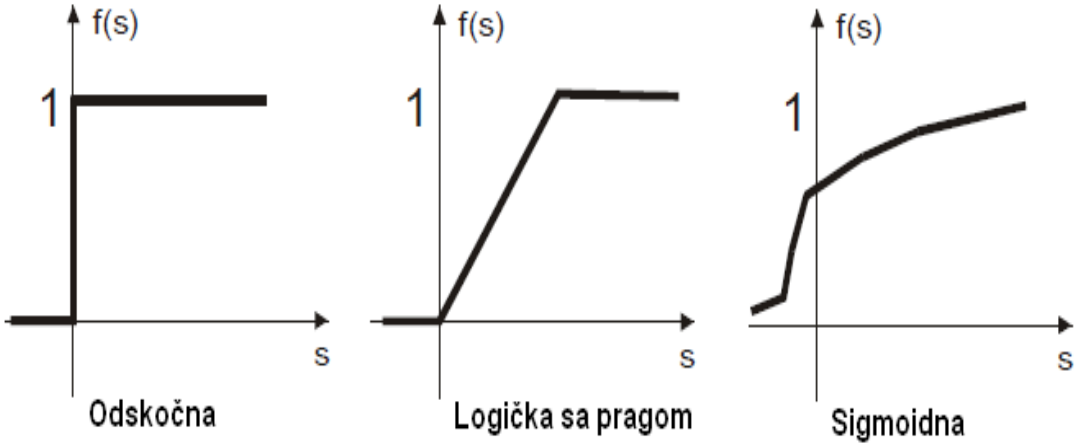


Model neuronske mreže sa tri sloja

Gotovo svaka nelinearna funkcija može da se koristi kao *prenosna funkcija*, mada se za backpropagation algoritam najčešće koriste sigmoidne funkcije kao što su: logistička, arkustanges ili gausova funkcija.

Danas se u neuronskim mrežama najviše koriste tri tipa prenosnih funkcija (slika):

- Odskočna
- Logička sa pragom
- Sigmoidna



Učenje neuronskih mreža se svodi na učenje iz primera kojih bi trebalo da bude što je više moguće, da bi mreža mogla da se ponaša što preciznije u kasnijoj eksploataciji. Proces učenja dovodi do korigovanja sinaptičkih težina. Kada uzorci koji se predstavljaju mreži ne dovode više do promene ovih koeficijenata, smatra se da je mreža obučena. Postoje tri tipa obučavanja neuronskih mreža:

- *Nadgledano obučavanje - mreži se predstavljaju ulazni podaci i očekivani izlazni podaci,*
- *Obučavanje ocenjivanjem - mreži se ne predstavljaju očekivani izlazni podaci nego joj se posle određenog perioda predstavlja ocena prethodnog rada, i*
- *Samoorganizacija - mreži se predstavljaju isključivo ulazni podaci.*

Nadgledano obučavanje

Proces obučavanja neuronskih mreža započinje zadavanjem slučajnih vrednosti težinskih koeficijenata veza i dovođenjem oblika na ulazni sloj. Zatim se mreža aktivira i upoređuje se izlazni i zadati izlazni oblik. Obučavanje se vrši tako da se ažuriraju težinski koeficijenti sa ciljem da se u sledećoj iteraciji dobija izlaz koji je bliži zadatoj vrednosti. U trenutku kada se postigne zadovoljavajući rezultat sa jednim ulaznim oblikom, na ulaz se dovodi drugi, itd.

Kada se završi sa svim oblicima iz obučavajućeg skupa, na ulaz mreže se dovodi ponovo prvi ulazni oblik. Ova se procedura nastavlja sve dotle dok se ne dođe do zadovoljavajućih rezultata za sve oblike iz obučavajućeg skupa. Jednom kada je obučavanje mreže završeno, težinski koeficijenti veza ostaju nepromenjeni. Tek sada kada je mreža obučena može se primeniti za predviđeni zadatak.

Nenadgledano obučavanje

Kod nenadgledanog obučavanja izlazna vrednost se ne upoređuje sa zadatom vrednošću. Ovom metodom mreža klasifikuje ulazne oblike u više grupa. Broj jedinica u izlaznom sloju odgovara broju različitih grupa – to znači da u jednom datom trenutku postoji samo jedan aktivan izlaz.

Realizacija neuronske mreže

Neuronska mreža može da se realizuje na dva načina:

- Hardverski i
- Softverski

Kod hardverske realizacije veštački neuroni su fizički međusobno povezani i oponašaju veze između bioloških neurona. Neuroni se realizuju kao jednostavna integrisana kola.

Kod softverske realizacije neuronske mreže se obično simuliraju na personalnim računarima, u kojima je veza između čvorova logička (virtuelna).

Prednost hardverske realizacije je to što može da koristi mogućnost paralelnog procesiranja informacija ukoliko se svakom neuronu u mreži dodeli po jedan procesor.
Prednost softverske realizacije neuronskih mreža na personalnom računaru je u tome što se lakše uspostavljaju i menjaju veze između pojedinih neurona u mreži.

U praksi se softverska realizacija koristi za testiranje, a konkretna realizacija koja se primenjuje u praksi može biti realizovana i hardverski čime se dobija na brzini.

Podela neuronskih mreža

Neuronske mreže se mogu realizovati na veliki broj načina, pa stoga postoji veliki broj načina njihove klasifikacije. Osnovna klasifikacija neuronskih mreža se može izvršiti prema:

- *Broju slojeva,*
- *Vrsti veza između neurona,*
- *Vrsti obučavanja neuronskih mreža,*
- *Prema smeru prostiranja informacija*
- *Prema vrsti podataka, itd.*

Jedna od najopštijih podela neuronskih mreža je prema broju slojeva, tako imamo:

- *Jednoslojne*
- *Višeslojne*

Danas se uglavnom izučavaju i primenjuju višeslojne neuronske mreže koje pored ulaznih i izlaznih slojeva, sadrže neurone na srednjim (skrivenim) slojevima.

Neuronske mreže se mogu podeliti prema vrstama veza odnosno arhitekturi na:

- Slojevite – neuroni su raspoređeni tako da formiraju slojeve. Na ulazu jednog neurona se dovode izlazi svih neurona sa predhodnog sloja, a njegov izlaz se vodi na ulaze svih neurona na narednom sloju. Neuroni sa prvog (ulaznog) sloja imaju samo po jedan ulaz. Izlaz neurona sa zadnjeg (izlaznog) sloja predstavljaju izlaze mreže. Predstavnik: *backpropagation algoritam*.
- Potpuno povezane – izlaz jednog neurona se vodi ka ulazu svih neurona u mreži. Predstavnik: *Hopfildova neuronska mreža*.
- Celularne – međusobno su povezani samo susedni neuroni. Bez obzira na lokalnu povezanost, signali se prostiru i na neurone i van susedstva zbog indirektnog prostiranja informacija. Predstavnik: *CNN – Cellular Neural Network*.

Podela neuronskih mreža prema vrsti obučavanja:

- Nadgledano obučavanje – *Supervised training*
- Delimično nadgledano obučavanje
- Nenadgledano obučavanje – *Unsupervised training*

Prema vrsti podataka koje obrađuju neuronske mreže se mogu podeliti na:

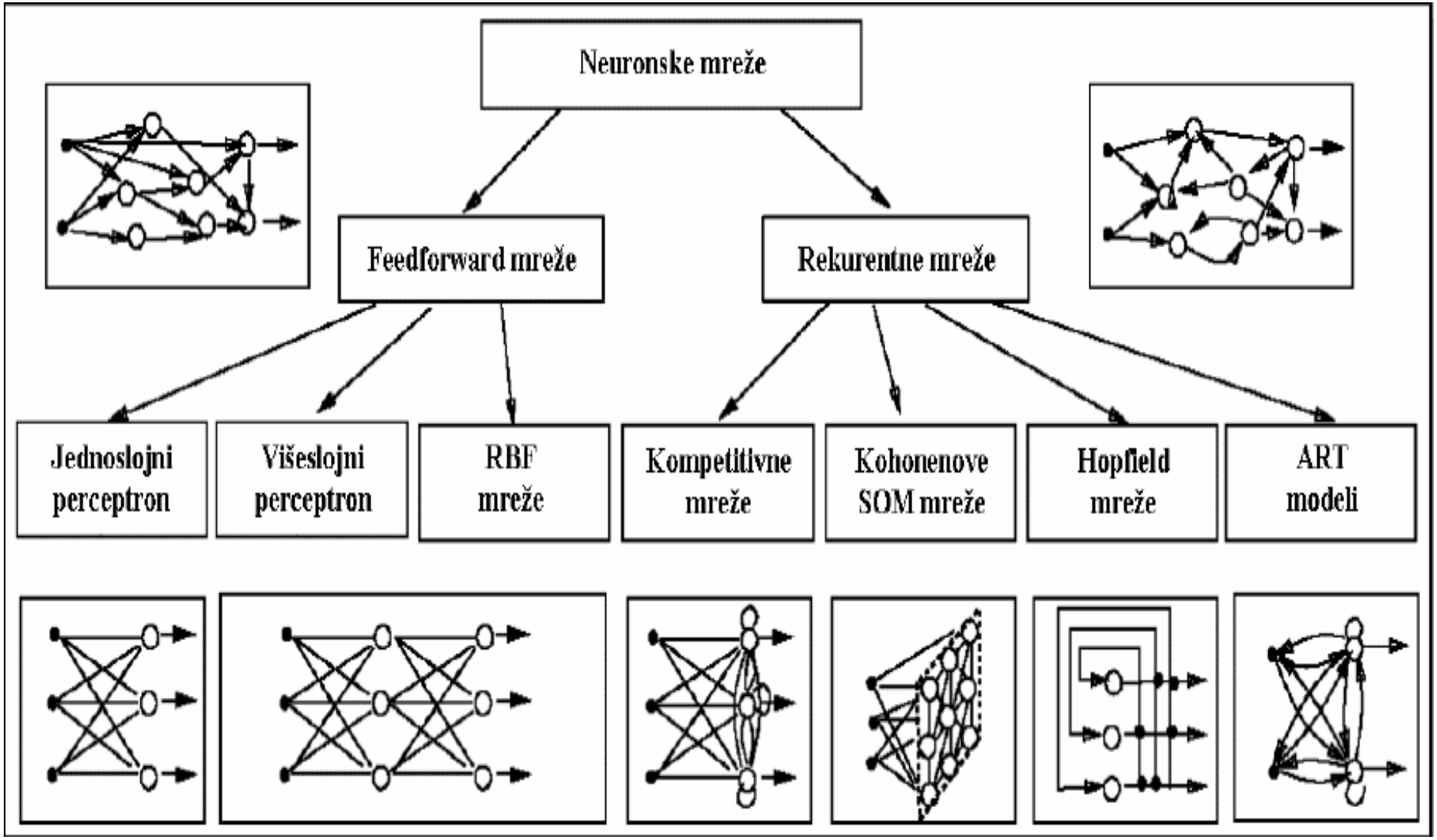
- Analogne
- Diskretne

Ova vrsta podele neuronskih mreža se retko koristi zato što su gotovo sve neuronske mreže diskretne.

Prema smeru prostiranja informacija kroz mrežu neuronske mreže se mogu podeliti na:

- Feedforward (nerekurzivne, nerekurentne ili nepovratne) – vrše prostiranje signala samo u jednom smeru (od ulaza prema izlazu) odnosno propagaciju signala. Predstavnici: *Višeslojni perceptron sa primenjenim backpropagation algoritmom*.
- Feedback (rekurzivne, rekurentne ili povratne) – viši slojevi vraćaju informacije u niže slojeve. Izlaz iz neurona se vraća u niže slojeve ili u isti sloj. Predstavnici: *Hopfildove, Cellularne, Kohonenove, dvostruko asocijativne neuronse mreže*. Ove mreže imaju mnogo veće procesne sposobnosti u odnosu na feedforward mreže.

Vrste neuronskih mreža

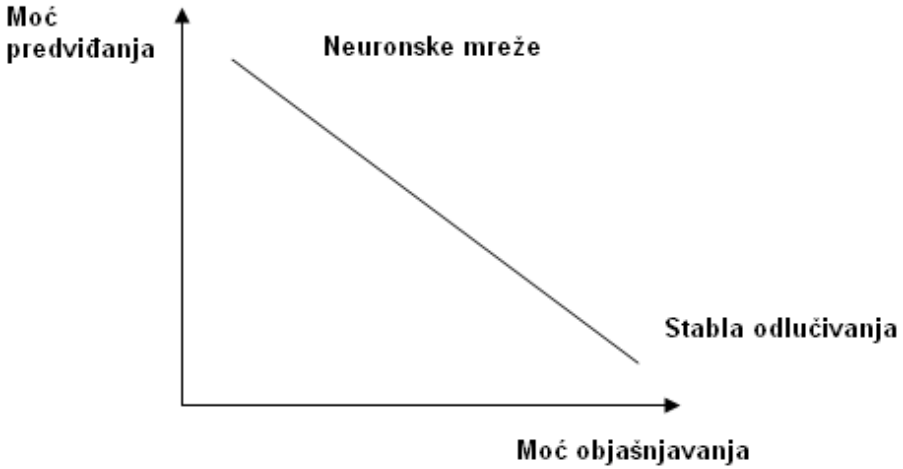


Prednosti, nedostaci i ograničenja neuronskih mreža

Prednost neuronskih mreža je njihova *sposobnost da uče*, koja im daje prirodniji interfejs ka modeliranju realnog sveta, u odnosu na klasične sisteme koji moraju biti programirani. Neuronske mreže su u stanju da pronađu veze izmedu pojava koje izmiču ljudskom intelektualnom aparatu potpomognutom klasičnim softverskim alatima.

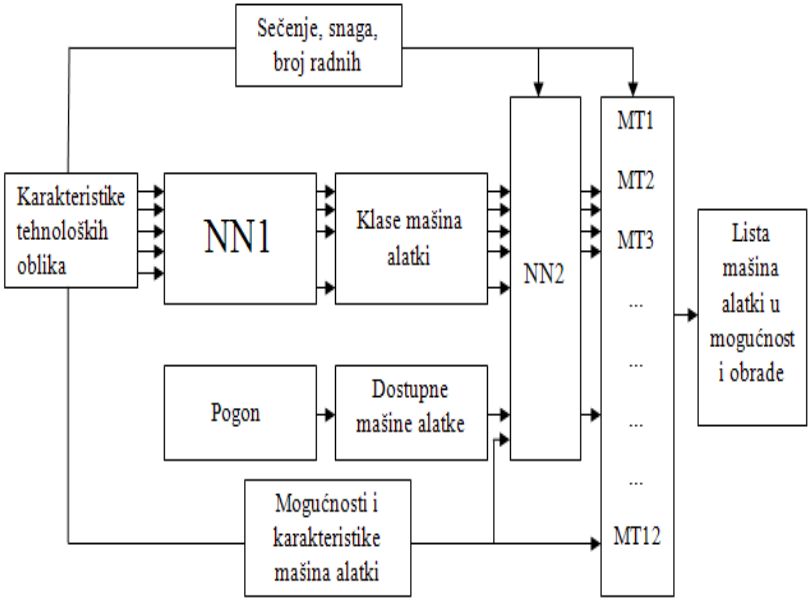
Neuronske mreže imaju i mogućnost *tolerancije nedostataka* – to znači da se mreža sastoji od više elemanata procesiranja, pa može da funkcioniše i ako dode do oštećenja dela mreže. Sposobne su da generalizuju, pa ako im se prezentuje nekompletan skup ulaznih podataka, mreža će ipak biti u stanju da dà izlaz.

Neuronske mreže ne rade dobro ono što ni ljudi ne rade dobro. Nisu dobre za aritmetičke proračune i zadatke obrade podataka. Iako imaju odličnu moć predviđanja, imaju slabu sposobnost objašnjavanja. Na slici je prikazan dijagram zavisnosti moći predviđanja i objašnjavanja neuronskih mreža.

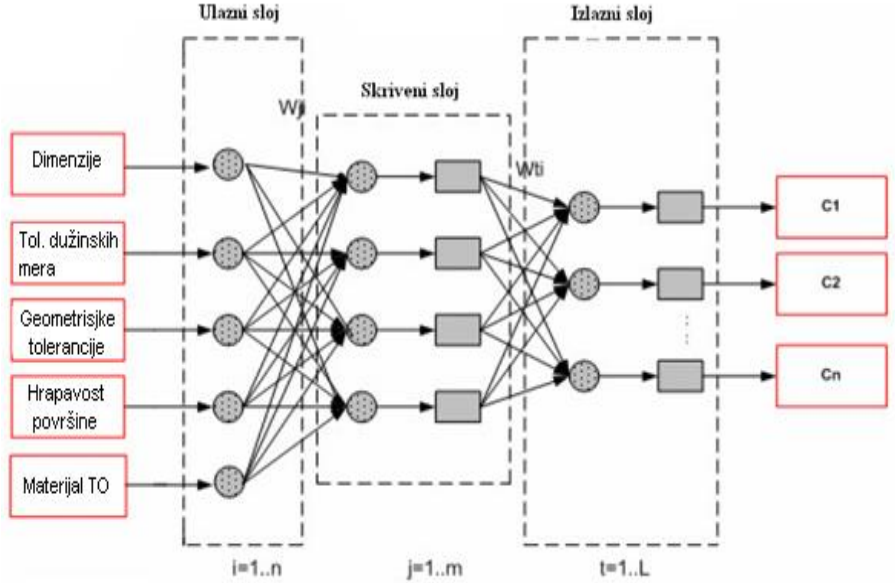


Sa dijagrama se vidi da neuronske mreže odlično predviđaju, a slabo objašnjavaju što je potpuno suprotno od npr. stabla odlučivanja. Neuronska mreža ne može da objasni korisniku kako je došla do određenog rešenja.

Današnji CAPP sistemi zasnovani na znanju, treba da obezbede i sposobnost učenja sistema, tako da se veštačke neuronske mreže koriste kao adekvatna računarska paradigma za njihov razvoj. Dakle, primenom neuronskih mreža CAPP sistemi mogu postati adaptivni i sposobni za učenje. Primena neuronskih mreža u proizvodnom mašinstvu posebno je izražena u oblasti modeliranja, upravljanja, klasifikacije i prepoznavanja skupova, grupisanja, procesiranja signala i optimizacije. Neuronske mreže se primenjuju u rešavanju različitih zadataka u okviru razvoja CAPP sistema i njihove integracije sa drugim CAx sistemima i aktivnostima poslovnog sistema. Neke od aktivnosti primene su u prepoznavanju tipskih tehnoloških oblika (feature), optimizaciji izbora zahvata i njihovog redosleda izvođenja, mašina, alata, pribora, parametara obrade, i dr.



Optimizacija izbora mašina alatki preko neuronskih mreža



Struktura neuronske mreže NN1

Napomena: pogledati odgovarajuću vežbu

Fuzzy logika

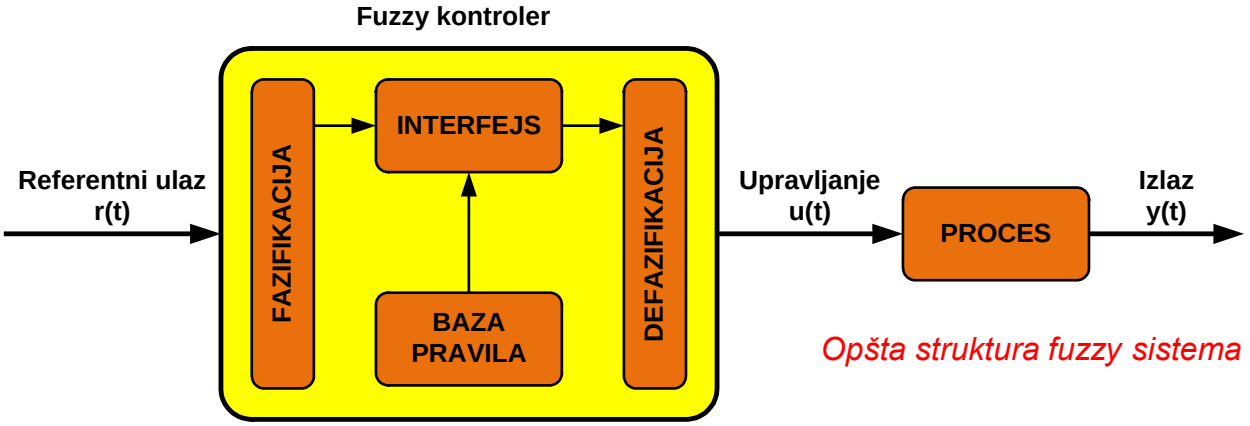
Postoje situacije u kojima nije moguće znanje o sistemu reprezentovati na apsolutno precizan način. Čak je više situacija u kojima moramo da koristimo neprecizne konstatacije. Na primer, «Marko je *visok* čovek», «Onaj automobil se približava *jako velikom* brzinom» su neprecizne rečenice a ipak ih svakodnevno koristimo.
Da bismo bili u stanju reprezentovati znanje o ovakvim sistemima (a ima ih *jako puno*) moramo da se odreknemo klasične (binarne) logike u kojoj je nešto ili tačno ili netačno (crno ili belo) i da koristimo fuzzy logiku (sve je nijansa sive boje).

Ulazi i izlazi mogu imati različite lingvističke nazive. Uobičajeno se promenjive nazivaju opisnim imenima, poput: nivo vode, priliv vode, ljudi srednjeg rasta, velike zarade, brzi automobili, mala rastojanja itd. Transformaciju ovakvih izraza u oblik matematičke predstave omogućava nam teorija *fuzzy* skupova.
Lingvističke promenjive bi trebalo da imaju i lingvističke vrednosti. To mogu biti: „negativno veliko“, „negativno srednje“, „negativno malo“, „blisko nuli“, „pozitivno veliko“, „dobro“, „otvori brzo“ i sl. Ovim vrednostima možemo da dodelimo i numeričku predstavu u cilju lakšeg i kraćeg obeležavanja.

Tako, veći *deo procesa odlučivanja u projektovanju tehnoloških procesa* se odvija u okruženju gde ciljevi i ograničenja često nisu i ne mogu biti precizno definisani. Zbog toga se zahteva određena aproksimacija u cilju dobijanja kvalitetnog modela realnog sistema, u čemu značajnu ulogu imaju tehnike *fuzzy teorije*, odnosno *fuzzy logike*.

Fuzzy upravljanje

Za razliku od formalne logike u kojoj se rezonovanje vrši sa dve vrednosti (tačno-netačno,0-1), fuzzy logika koristi brojeve iz intervala [0,1], što je mnogo bliže realnosti, ljudskom razmišljanju i izražavanju. Fuzzy upravljanje obezbeđuje formalnu metodologiju za predstavljanje, manipulaciju i implementaciju ljudskog heurističkog predznanja o tome kako kontrolisati jedan, određeni sistem. Osnovni cilj fuzzy pristupa je da, umesto da jezikom matematike pokuša da reši problem upravljanja sistemom, omogući implementaciju inženjerskog iskustva o procesu u sam algoritam kontrolera.



Opšta struktura fuzzy sistema

Baza pravila - sadrži znanje o tome kako najbolje kontrolisati sistem, i to u formi skupa logičkih pavila (*if – then*).

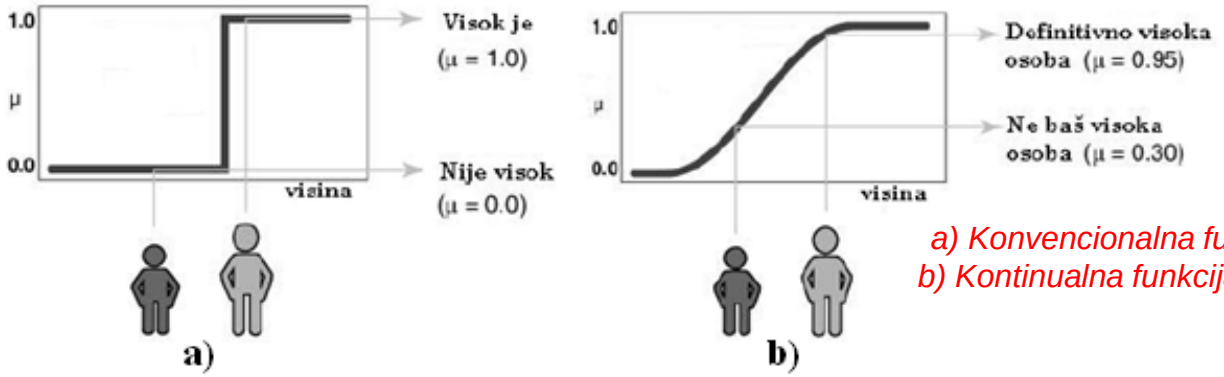
Interfejs - je mehanizam za procenjivanje koja kontrolna pravila su relevantna za trenutno stanje sistema i odlučuje logičkim sklopom kakav će biti upravljački signal, tj. ulaz u proces.

Fazifikacija – modifikuje signale ulaza tako da mogu biti pravilno protumačeni i upoređeni sa pravilima u bazi pravila. Crisp signal pretvara u adekvatan fuzzy oblik.

Defazifikacija – transformiše zaključak interfejsa u takav oblik signala, da ovaj može biti signal koji predstavlja ulaz u proces. Ovo je transoformacija fuzzy oblika u crisp oblik signala, koji je ,,razumljiv'' procesu.

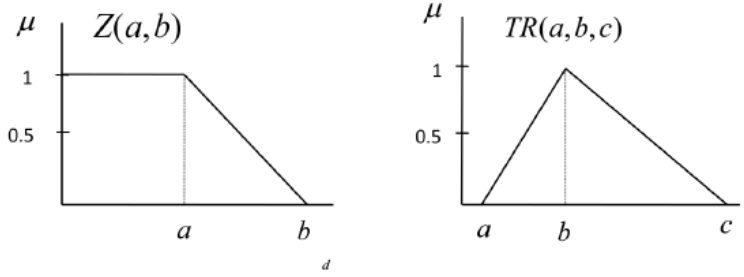
Funkcije pripadanja

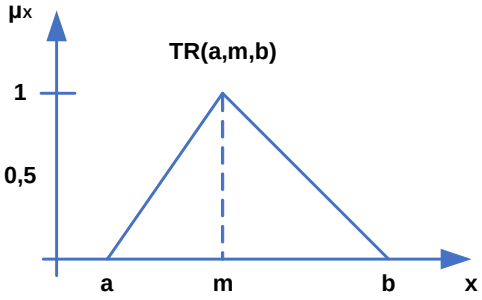
Funkcije pripadanja ilustruju prirodu lingvističkih vrednosti. Funkcija pripadanja predstavlja kontinualno merilo sigurnosti da li je naša promenljiva klasifikovana kao ta lingvistička vrednost. *Ova funkcija određuje stepen pripadanja nekog objekta datom fuzzy skupu.*

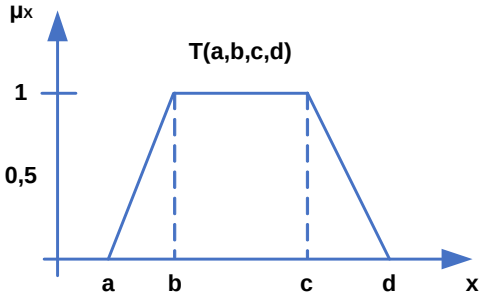


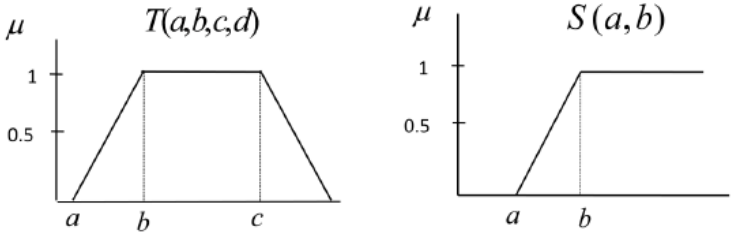
a) Konvencionalna funkcija pripadanja skupu visokih osoba
b) Kontinualna funkcija pripadanja fuzzy skupu visokih osoba

Ove funkcije mogu da imaju različite oblike. Kakva će biti funkcija zavisi od uslova i ponašanja sistema.







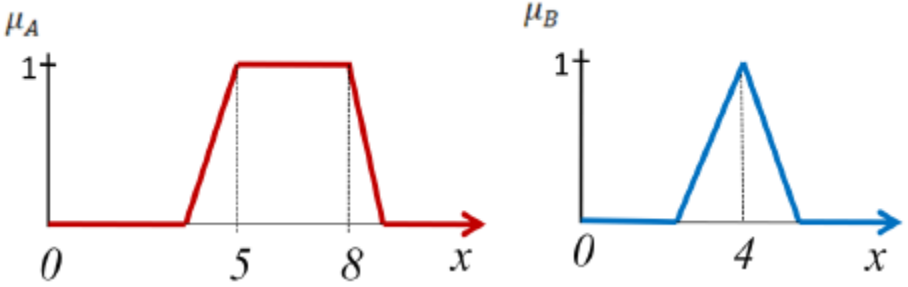


$$\mu_A(x) = \begin{cases} 0, x \le a, x \ge b \\ \dfrac{(x-a)}{(m-a)}, x \in (a,m] \\ \dfrac{(b-x)}{b-m}, x \in (m,b) \end{cases}$$

$$\mu_A(x) = \begin{cases} 0, x \le a, x \ge d \\ \dfrac{(x-a)}{(b-a)}, x \in (a,b) \\ 1, x \in (b,c) \\ \dfrac{(d-x)}{(d-c)}, x \in (b,d) \end{cases}$$

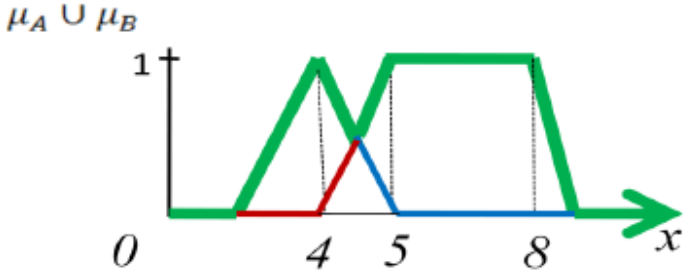
Logički operateri kod Fuzzy brojeva

Ukoliko postoji više uslova pravila, ukupno zadovoljenje uslova se računa preko operatora nad fuzzy funkcijama pripadnosti. Slično kao i kod klasičnih (*crisp*) *skupova* i kod *fuzzy* skupova su definisane operacije: *unija (*logički operator *ili)*, *presek (*logički operator *i)* i *negacija*.



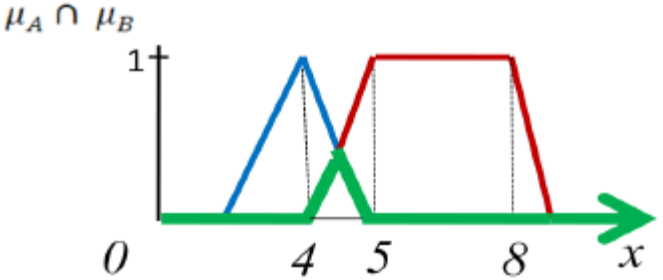
Skup fuzzy brojeva

Logički operator *ili* je kod fuzzy skupova definisan kao funkcija *max:* $\mu 1 \vee \mu 2 = \max\{\mu 1, \mu 2\}$



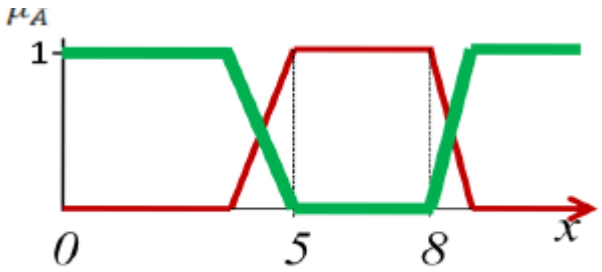
Slika – Unija skupa (funkcija max)

Logički operator *i* je kod fuzzy skupova definisan kao funkcija *min:* $\mu 1 \wedge \mu 2 = \min\{\mu 1, \mu 2\}$



Slika – Presek skupa (funkcija min)

Negacija je kod fuzzy skupova definisana kao: $\overline{\mu 1} = 1 - \mu 1$



Slika – Operator negacije(1-μ)

Primer: Određivanje vrednosti automobila

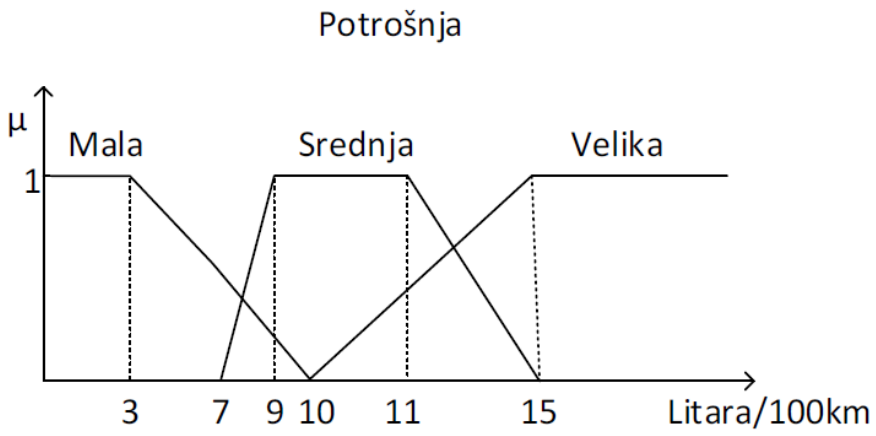
a) Uz pomoć fazi logike kreirati model koji će određivati vrednost automobila na osnovu njegovih karakteristika: Potrošnja i Pouzdanost.

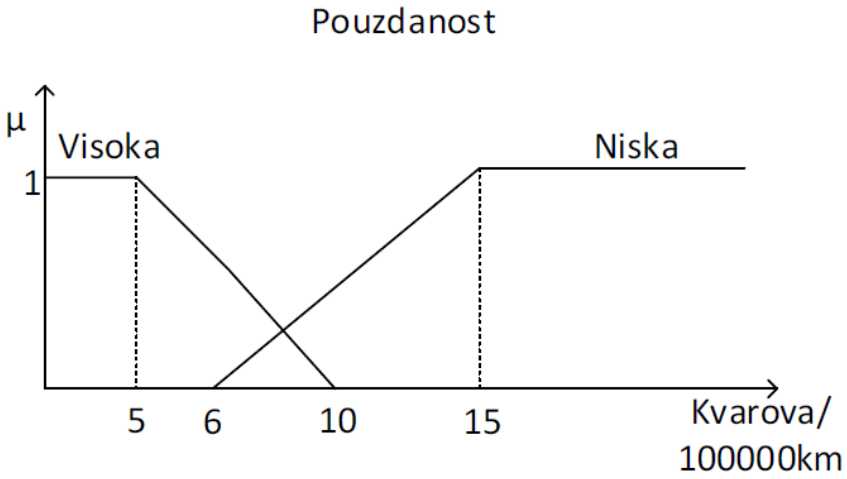
b) Odrediti vrednosti automobila koji imaju sledeće karakteristike:

Automobil	Potrošnja	Pouzdanost
VW	9	8
Dacia	12	14
BMW	9	5
Volvo	15	0

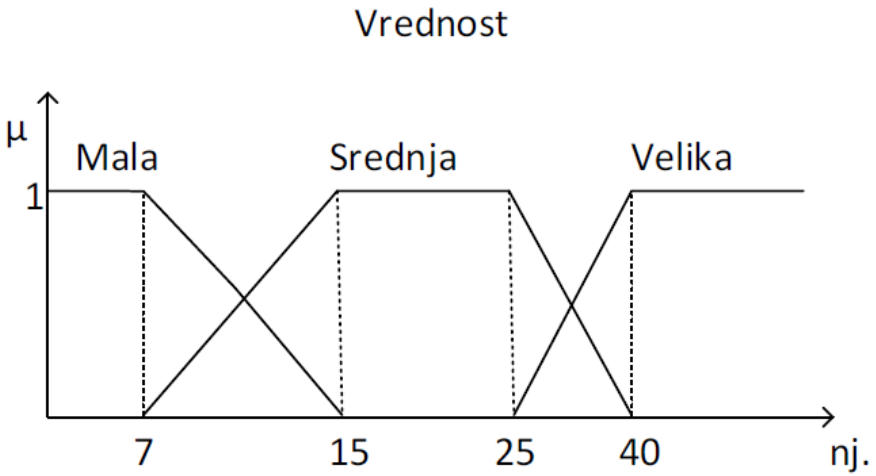
Rešenje

a) Prvo se vrši definisanje Fuzzy vrednosti ulaznih promenljivih Potrošnja i Pouzdanost





Zatim definišemo Fuzzy vrednost izlazne promenljive Vrednost



Potrošnja	
Mala	Z(3, 10)
Srednja	T(7, 9, 11, 15)
Visoka	S(10, 15)
Pouzdanost	
Visoka	Z(5, 10)
Niska	S(6, 15)
Vrednost	
Mala	Z(7, 15)
Srednja	T(7, 15, 25, 40)
Visoka	S(25, 40)

Na kraju na osnovu inženjerskog znanja definišemo skup uzročno posledičnih pravila:

	Potrošnja	Pouzdanost	Vrednost
1.	Mala	Niska	Srednja
2.	Mala	Visoka	Velika
3.	Srednja	Niska	Srednja
4.	Srednja	Visoka	Velika
5.	Velika	Niska	Mala
6.	Velika	Visoka	Srednja

Time je kreiran **Fuzzy model za određivanje vrednosti vozila**:

Napomena:
Svako od pravila se takođe može predstaviti i u obliku „ako-onda“ pravila.
Tako se drugo pravilo može interpretirati kao:

AKO **Potrošnja** je ***Mala*** i **Pouzdanost** je ***Visoka*** ONDA Vrednost je ***Velika***

b) Kada imamo definisane funkcije pripadnosti za ulazne i izlazne promenljive, i skup uzročno posledičnih pravila koja povezuju te promenljive možemo da odredimo vrednost izlazne promenljive za novo pojavljivanje.

Znamo da automobil VW troši 9l na 100 km i da ima 8 kvarova na 100000 km. Sada uz pomoć pravila proporcije određujemo vrednost funkcije pripadnosti za obe promenljive.

Potrošnja:

- Velika: 0
- Srednja: (9-7)/(9-7) = 1
- Mala : (10-9)/(10-3) =0.143

Pouzdanost:

- Visoka: (10-8)/(10-5) = 0.4
- Niska: (8-6)/(15-6) = 0,222

Zatim prelazimo na proces zaključivanja na osnovu definisanih pravila. Prvo u tabeli uočavamo koja se pravila odnose na naše novo pojavljivanje (aktivirana pravila):

Pravilo	Potrošnja	Pouzdanost	Vrednost
1.	Mala (0.143)	Niska (0.222)	Srednja (0.143)
2.	Mala (0.143)	Visoka (0.4)	Velika (0.143)
3.	Srednja (1)	Niska (0.222)	Srednja (0.222)
4.	Srednja (1)	Visoka (0.4)	Velika (0.4)
5.	Velika (0)	Niska (0.222)	Mala (0)
6.	Velika (0)	Visoka (0.4)	Srednja (0)

Na osnovu pravila i vrednosti funkcija pripadnosti za ulazne promenljive, računamo vrednosti funkcije pripadnosti za sve dopuštene vrednosti izlazne promenljive. Iste se računaju kao ukupan nivo zadovoljenja pojedinačnih uslova pravila:

Pravilo 1: Srednja vrednost 0.143 /\ 0.222 = 0.143
Pravilo 2: Velika vrednost: 0.143 /\ 0.4 = 0.143
Pravilo 3: Srednja vrednost 1 /\ 0.222 = 0.222
Pravilo 4: Srednja vrednost: 1 /\ 0.4 = 0.4

Kako imamo dva pravila koja nam daju pripadnost za vrednost *srednja vrednost* (pravila 1 i 3) i za *visoka vrednost* (pravila 2 i 4), mora se odrediti koja će se vrednost koristiti:

Srednja vrednost 0.143 V 0.222 = 0.222
Velika vrednost: 0.143 V 0.4 = 0.4

Konačno, dolazimo do **faze defazifikacije** gde lingvističke vrednosti promenljive Vrednost (velika i srednja) prevodimo u jednu preciznu (matematičku) vrednost koja zapravo predstavlja konačnu ocenu vrednosti novog pojavljivanja (u našem slučaju automobila).

$$DFV = \frac{\sum [MV_i \cdot FMV_i]}{\sum [MV_i]}$$

- DFV – Defazifikovana fazi vrednost
- MVi – Koeficijent pripadnosti i-tom zaključku
- FMVi – Reprezentativna vrednost i-tog zaključka (fuzzy skupa). Za S i Z oblike funkcije se uzima vrednost na granici sa potpunom pripadnošću skupu (μ=1), dok se za T oblik uzima središnja vrednost od svih sa potpunom pripadnošću skupu. Uzimaju se vrednosti koje su obeležene crvenom bojom
Z(a,b), S(a,b), T(a, b, c, d) i Tr(a, b, c)
- N – broj fuzzy zaključaka

VREDNOST VW= (0.222*(15+25)/2+0.4*40)/(0.4+0.222)=32.857

Drugi automobil, Dacia, troši 12 l na 100 km i ima 14 kvarova na 100000 km. Prvo računamo vrednosti funkcije pripadnosti za potrošnju i pouzdanost.

Potrošnja:

- Mala: 0
- Srednja: (15-12)/(15-11) = 0.75
- Velika: (12-10)/(15-10) = 0.4

Pouzdanost:

- Visoka: 0
- Niska: (14-6)/(15-6) = 0.889

Sledeća pravila su aktivirana.

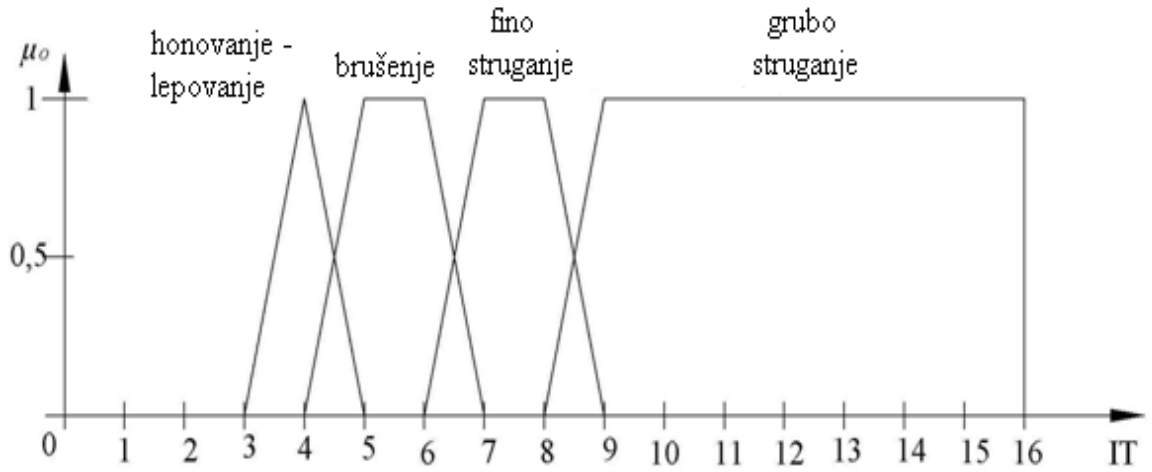
Pravilo	Potrošnja	Pouzdanost	Vrednost
1.	Mala (0)	Niska (0.889)	Srednja (0)
2.	Mala (0)	Visoka (0)	Velika (0)
3.	Srednja (0.75)	Niska (0.889)	Srednja (0.75)
4.	Srednja (0.75)	Visoka (0)	Velika (0)
5.	Velika (0.4)	Niska (0.889)	Mala (0.4)
6.	Velika (0.4)	Visoka (0)	Srednja (0)

Kako nemamo više pravila koja imaju pripadnost za istu vrednost onda prelazimo direktno na fazu defazifikacije.

$$\text{VREDNOST} = (0.4 * 7 + 0.75 * (15 + 25)/2)/(0.4 + 0.75) = 15.478$$

Primer primene fuzzy logike kod definisanja tehnološkog procesa obrade

Primer primene ***fuzzy logike*** prikazan je za određivanje fuzzy promenljive klasa kvaliteta, odnosno tolerancija. U literaturi je definisan ekomičan kvalitet površina za pojedine vrste obrade, kao što su: struganje, glodanje, cilindrično brušenje, ravno brušenje, honovanje, lepovanje, bušenje, proširivanje, razvrtanje. Funkcije pripadnosti fuzzy promenljivih kvaliteta obrađene površine, odnosno klasa tolerancija, za struganje, brušenje i honovanje – lepovanje mogu se definisati na način kao što je to prikazano na slici.



Funkcije pripadnosti klasa tolerancija za pojedine obrade

$$\mu_{BR}(IT)=\begin{cases} o; & IT\le 4 \\ IT-4; & 4<IT\le 5 \\ 1; & 5<IT\le 6 \\ -IT+7; & 6<IT\le 7 \end{cases} \qquad \mu_{GS}(IT)=\begin{cases} o; & IT\le 8 \\ IT-8; & 8<IT\le 9 \\ 1; & 9<IT\le 16 \\ 0; & 16<IT \end{cases}$$

$$\mu_{FS}(IT)=\begin{cases} o; & IT\le 6 \\ IT-6; & 6<IT\le 7 \\ 1; & 7<IT\le 8 \\ -IT+9; & 8<IT\le 9 \end{cases} \qquad \mu_{HON}(IT)=\mu_{LEP}(IT)=\begin{cases} 0; & IT\le 3 \\ IT-3; & 3<IT\le 4 \\ -IT+5; & 4<IT\le 5 \end{cases}$$

Gde su:

- *BR – obrada brušenjem*
- *FS – obrada finim struganjem*
- *GS – obrada grubim struganjem*
- *HON – obrada honovanjem*
- *LEP – obrada lepovanjem*

Agent bazirane tehnologije

Agenti predstavljaju softver koji ima sposobnosti da samostalno, bez intervencije korisnika, izvršava postavljeni zadatak i izveštava korisnika o završetku zadatka ili pojavi očekivanog događaja. Agent je računarski sistem, koji u interakciji sa okruženjem, ima sposobnost da fleksibilno i samostalno reaguje u skladu sa ciljevima koji su mu postavljeni. U ovoj definicija ističe se tri ključna zahteva: *interakcija sa okruženjem, autonomnost i fleksibilnost*.
Interakcija sa okruženjem u ovom kontekstu znači da je agent sposoban da reaguje na ulaz dobijen iz okruženja i da može da izvodi akcije koje menjaju okruženje u kome agenti deluju. *Autonomnost* znači da je sistem u stanju da reaguje bez intervencije korisnika (ili drugih agenata), i da ima kontrolu nad sopstvenim akcijama. Takav sistem bi trebao da bude sposoban da uči iz iskustva.

Bez obzira na oblast primene agenata, oni uvek imaju tri osnovna dela:

- *Bazu znanja, koja sadrži podatke i domene znanja neophodne da agent uspešno obavlja svoje aktivnosti,*
- *Koordinacionu jedinicu, koja vrši kontrolu interakcije sa dugim agentima, uključujući interkomunikaciju, pregovore, koordinaciju i saradnju, i*
- *Jedinica za rešavanje problema, koja obuhvata nezavisno učenje, planiranje, rasuđivanje, donošenje odluka za izvršavanje određenih aktivnosti i zadataka.*

Postoje različite vrste agenata u zavisnosti od :

- *Ponašanja,* reaktivni (reactive) i savetodavni (deliberative) agenti,
- *Funkcionalnosti,* interfejs (interface) i internet (internet) agenti,
- *Mobilnosti,* mobilni (mobile) i stacionarni (stationary) agenti, i
- *Strukture,* logički ili softverski i fizički ili hardverski agenti.

Sistemi u kojima je upotrebljeno više agenata radi rešavanja zajedničkog problema nazivaju se multi agentni sistemi – MAS (multy-agent system). U ovakvim sistemima neophodno je da agenti imaju mogućnost međusobne komunikacije (LAN, Internet, itd.) u cilju razmene iskustva ili "pregovaranja" i dobijanja optimalnog rešenja.
Agenti koji se upotrebljavaju u multi agentnim sistemima mogu biti jednaki po karakteristikama ili se mogu razlikovati prema specijalnostima. Multi agentni sistemi su idealni za predstavljanje problema koji imaju više različitih metoda za rešavanje problema, višestruke perspektive i/ili višestruke entitete rešavanja problema. Omogućavaju izradu paralelnih računarskih sistema, pomažu pri radu sa vremenski ograničenim rezonovanjem i robusnim sistemima – ako su odgovornosti podeljene.
U sistemima izrađenim na ovaj način umesto da procesom upravlja jedan kompleksan agent, upravljanje se deliti na više agenata koji prema svojim specijalnostima preuzimaju nadležnost nad kontrolom složenog procesa.
Upotrebom multi agentnih sistema se povećava bezbednost sistema u situacijama otkazivanja jednog od agenata, čitav sistem može biti automatski rekonstruisan ili zaustavljen na kontrolisan način.

Prilikom projektovanja multi agentnih sistema potrebno je definisati broj agenata, kritičnu količinu vremena za obavljanje zadatka, dinamiku pristizanja ciljeva, troškove komunikacije, cenu neuspeha, uticaj korisnika, neodređenost okruženja. Na nivou svakog agenta potrebno je definisati: početna stanja u domenu, moguće akcije drugih agenata, izlazne akcije agenta. Sa povećanjem broja agenata koji sarađuju na rešavanju zajedničkog problema javljaju problemi kao što su

- *Kooperativnost (dizajnirati agente tako da zajednički rade na zajedničkim ciljevima),*
- *Koordinacija (upravljati agentima tako da si izbegnu štetne interakcije a korisne interakcije iskoriste),*
- *Pregovaranje (dolaženje do dogovora koji su prihvatljivi svim objektima/agentima koji učestvuju u rešavanju problem).*

Primer primene agent tehnologije u razvoju CAPP sistema

Sistem CoCAPP (Cooperative Computer-Aided Process Planning), predstavlja primer CAPP sistema koji je zasnovan na kooperativnim agentima. U okviru ovog sistema agenti su zapravo ekspertni sistemi, sa sličnom osnovnom strukturom za čiji razvoj je korišćena ista ekspertna ljuska, koji se preko funkcionalnih adaptera prilagođavaju različitim zadacima koje imaju u sistemu. U sistemu postoje tri tipa agenata:

- *D-agent,* koji ima zadatak uvoza CAD modela (STEP format) i prosleđivanja B-agentu, kao i prezentaciju rešenja korisniku.
- *B-agent,* koji ima ulogu obezbeđivanja komunikacije između P-agenata, kroz četiri odvojena prostora podataka: problem, predlog, konflikt i rešenje.
- *P-agent,* koji sprovodi specifične zadatke u projektovanju tehnoloških procesa (tehnološko prepoznavanje, izbor mašina, alata, pribora, procenu troškova, itd.).