Pomoć pri odabiru programskog jezika uz pomoć dijagrama utjecaja

Josip Užarević

Ekspertni sustavi

Fakultet elektrotehnike i računarstva
Sveučilište u Zagrebu

Siječanj 2017

Sadržaj

1 Uvod

Pri započinjanju implementacije nekakvog projekta, bilo u akademske svrhe ili u profesionalnom smislu, student ili radnik katkad će se zapitati: *Koji programski jezik upotrijebiti? S obzirom na moje predznanje, hoće li mi odabir X uzeti previše vremena? Budući da se radi o arhitekturi Klijent-Poslužitelj, koji mi je programski jezik za poslužiteljsku aplikaciju pametno odabrati?* Zaista, u moru programskih jezika koji su danas u upotrebi, teško je odabrati onaj koji će se programeru ispostaviti kao najisplativiji / najlakši / najupotrebljiviji ili nešto drugo.

U tu svrhu, u okviru projekta na kolegiju *Ekspertni sustavi*, javila se ideja za ostvarenje ekspertnog sustava koji će davati pomoć pri ovom, teškom odabiru. Ekspertni sustav je upravo odličan odabir za rješavanje ovog problema, jer je moguće direktno iskoristiti ekspertno (samim time moguće i subjektivno) znanje u sustav, a uz to dobiti i *ne-diskretne* odgovore (umjesto odgovora *DA/NE*, dobivaju se vjerojatnosni odgovori).

2 Ekspertno znanje

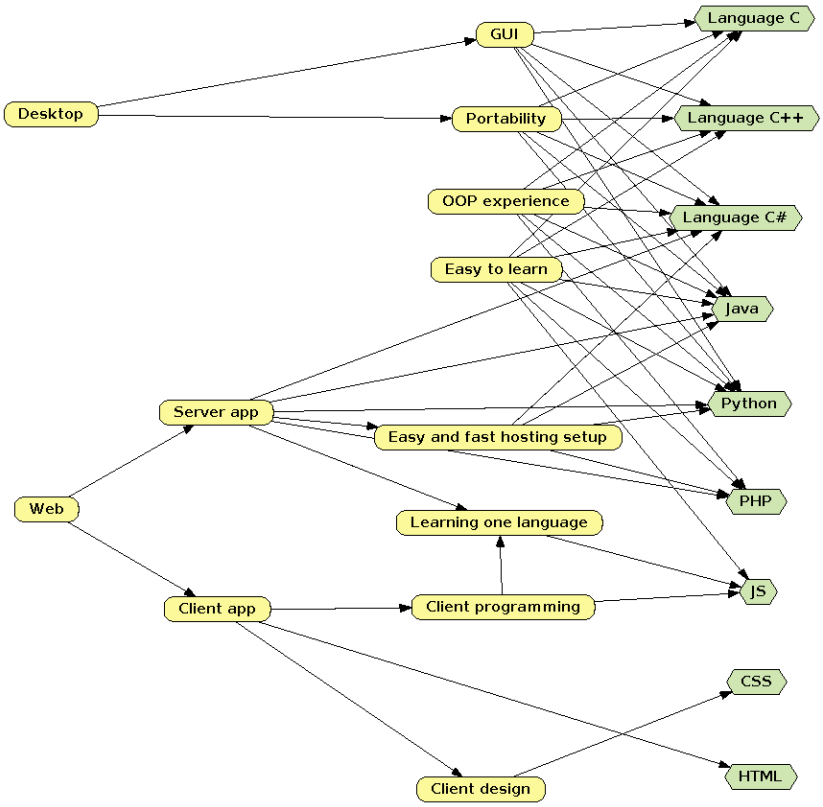
Ekspertno znanje je stečeno isključivo iskustvom autora ovog rada, koje je pak stečeno dugogodišnjom praksom, kako u fakultetske, tako i u profesionalne svrhe te praćenjem *programerske scene*. Samim time raste vrijednost ubačenog ekspertnog znanja, budući da u rad nisu ubačene nekakve gotove vrijednosti dohvaćeni s raznih web-portala ili slično.

3 Model ekspertnog sustava

Za implementaciju ekspertnog sustava odabran je model *mreže utjecaja*, koja modelira upravo međusobne utjecaje varijabli te, kao (relativno) završne čvorove može sadržavati i čvorove korisnosti. U danom modelu, varijable koje korisnik može unositi su ustvari *čvorovi izglednosti*, dok su takoreći izlazne vrijednosti upravo *čvorovi korisnosti*. Izgled mreže prikazan je na slici 1.

Svakom čvoru izglednosti su definirane uvjetne vjerojatnosti za njegove *roditeljske čvorove*. Prvotno su i čvorovima korisnosti bile dodjeljivane tablice vjerojatnosti, no kako neki čvorovi korisnosti imaju čak 6 ulaza, tako su njihove tablice vjerojatnosti imale $2^6 = 64$ unosa. Zbog ovoga nastaje problem, ne samo u količini posla prilikom unosa, nego još više u unošenju šuma u cijeli sustav. Naime, vrlo teško ekspert može pretpostavljati vjerojatnosti za toliki broj kombinacija samo za jedan čvor, a da ne unese kontradiktorne podatke, kako unutar samog čvora, tako i unutar cijele mreže.

Stoga, čvorovi korisnosti su izvedeni kao linearna kombinacija svih ulaza. Za svaki ulaz ekspert unosi *ocjenu* od -5 do -5, pri čemu je -5 ekvivalentno izjavi *ulaz negativno utječe na izbor ovog jezika*, +5 *ulaz pozitivno utječe na izbor ovog jezika*, dok je unos nule jednak izjavi *ulaz ne utječe na izbor ovog jezika.* Ovakav izbor funkcije korisnosti se ustvari pokazao kao jako dobar, jer se ujedno



Slika 1: Model mreže utjecaja za problem određivanja programskog jezika za učenje

pri postupku zaključivanja sustava jasno vidi i raspon svakog izlaza te samim time i *apriori* preferenciju odabira jezika.

3.1 Unesene vrijednosti u model

Slijedi detaljan prikaz unesenih uvjetnih vjerojatnosti u čvorove izglednosti. Kod specifičnih slučajeva bit će navedeni razlozi i ishodi odabira upravo takvih vjerojatnosti. Čvorovi za koje nisu prikazane tablice su čvorovi bez roditelja - njihova razdioba je *apriori* uniformna. Mreža se ugrubo može podijeliti na tri dijela:

1. **Desktop dio** Ovaj dio mreže je zadužen za osnovno opredjeljenje ka desktop aplikacijama;
2. **Web dio** Ovaj dio mreže je zadužen za osnovno opredjeljenje ka web aplikacijama;
3. **Općeniti dio** Ovaj dio mreže je univerzalan, bez obzira na osnovno opredjeljenje.

3.1.1 Desktop dio

U tablici 1, ovakvim odabirom tablice, postignuto je to da ukoliko je odabran GUI, onda je nužno odabran i Desktop. Ako je odabran *NE*-GUI, tada je odabran i *NE*-Desktop. Ovime je jednoznačno određeno da je bespredmetno uzimati u obzir razvoj GUI-a ako se neće izrađivati Desktop aplikacija. Situacija je slična i u slučaju portabilnosti, kao i nekih drugih čvorova u mreži.

Desktop	Ne	Da
$P(\text{GUI} = \text{Da}\|\text{Desktop})$	0	0.5
$P(\text{GUI} = \text{Ne}\|\text{Desktop})$	1	0.5

Tablica 1: $P(\text{GUI}|\text{Desktop})$

Desktop	Ne	Da
$P(\text{Portability} = \text{Da}\|\text{Desktop})$	0	0.5
$P(\text{Portability} = \text{Ne}\|\text{Desktop})$	1	0.5

Tablica 2: $P(\text{Portability}|\text{Desktop})$

3.1.2 Web dio

U web dijelu osnovna podjela je na razvoj poslužiteljske i klijentske aplikacije, koja djelomično razdvaja neke daljnje čvorove (npr. *Easy and fast hosting setup*), dok se na nekim čvorovima stapaju *Learning one language*.

Web	Ne	Da
P(Server app = Da|Web)	0	0.5
P(Server app = Ne|Web)	1	0.5

Tablica 3: P(Server app|Web)

Web	Ne	Da
P(Client app = Da|Web)	0	0.5
P(Client app = Ne|Web)	1	0.5

Tablica 4: P(Client app|Web)

Slučaj prikazan u tablici 7 je zanimljiv, jer opisuje slučaj ako korisnik želi i poslužiteljski i klijentski dio programirati u jednom jeziku. Jezik *JavaScript* uz korištenje *frameworka Node.js* omogućava upravo to. Međutim, razvoj klijenta ne uključuje nužno programiranje (*JavaScript*) ukoliko se npr. radi statička web stranica. Stoga, ovom čvoru je najveća vjerojatnost ukoliko su odabrani klijentsko programiranje i poslužiteljska aplikacija. Ukoliko nijedan od ova dva čvora nije odabran, programiranje u jednom jeziku gubi smisao te je tada vjerojatnos jednaka nuli.

3.2 Korisnosti

Korisnosti odabira pojedinog programskog jezika, kao što je ranije rečeno, dobivene su unosom koeficijenata za svaki čvor koji je spojen s određenim programskim jezikom.

4 Proces zaključivanja

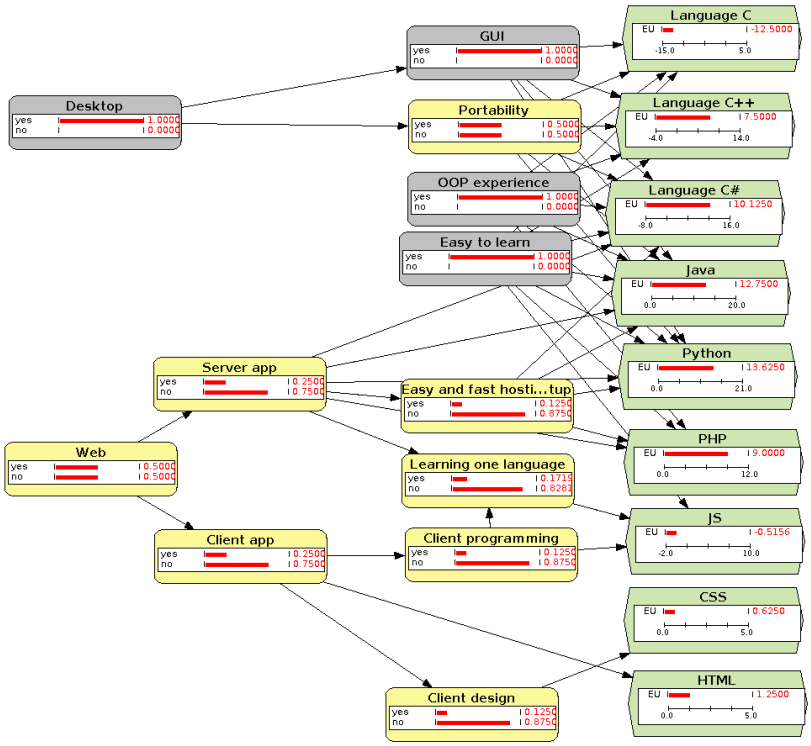
Na slici 2 prikazano je zaključivanje za primjer odabira Desktop GUUI aplikacije, s objektno orijentiranom paradigmom, da je jezik lako za naučiti te da je portabilnost nebitna. Najviše je bodovan Python, Java i C#, a najmanje C, što je i logično.

5 Programska podrška

Za realizaciju mreže korišten je alat OpenMarkov, nakon dugog traženja besplatnog programa, a da ima funkcionalnost modeliranja Bayesovih mreža i mreža utjecaja.

6 Zaključak

Izgradnja ekspertnog sustava pomoću Bayesovih mreža i njenih derivata je izuzetno zanimljiva, intuitivna te je proizvod naposljetku jednostavan za korištenje.



Slika 2: Primjer zaključivanja za odabrane čvorove *Desktop=Da, GUI=Da, OOP experience=Da, Easy to learn=Da*

Server app	Ne	Da
P(Easy and fast hosting setup = Da|Server app)	0	0.5
P(Easy and fast hosting setup = Ne|Server app)	1	0.5

Tablica 5: P(Easy and fast hosting setup|Server app)

Client app	Ne	Da
P(Client programming = Da|Client app)	0	0.5
P(Client programming = Ne|Client app)	1	0.5

Tablica 6: P(Client programming|Client app)

Međutim, ne postoji besplatan *state-of-the-art* program pomoću kojeg se može vršiti ovakvo modeliranje, a da program radi besprijekorno.

Client programming	Ne	Ne	Da	Da
Server app	Ne	Da	Ne	Da
$P(\text{Learning one language} = \text{Da} \mid \text{Client programming}, \text{Server app})$	0	0.5	0.5	0.75
$P(\text{Learning one language} = \text{Ne} \mid \text{Client programming}, \text{Server app})$	1	0.5	0.5	0.25

Tablica 7: $P(\text{Client programming} \mid \text{Client app})$

Client app	Ne	Da
$P(\text{Client design} = \text{Da} \mid \text{Client app})$	0	0.5
$P(\text{Client design} = \text{Ne} \mid \text{Client app})$	1	0.5

Tablica 8: $P(\text{Client design} \mid \text{Client app})$