Ekspertski sistemi

Lekcija 10: STRATEGIJE REŠAVANJA PROBLEMA

Strategije rešavanja problema

- iscrpno pretraživanje, t.j. ispitivanje svakog mogućeg stanja problema.
- Moguće je samo ako je prostor stanja razumno mali, negde do 10! Stanja (sistem XCON – ekspertski sistem za konfigurisanje VAX računara)
- Da bi se moglo raditi sa velikim prostorom stanja, koriste se dva principa:
  - Razvijanje efikasne metode koja može da radi sa velikim prostorom stanja
  - Transformacija prostora stanja u komponente koje se mogu zasebno obrađivati, na primer iscrpnim pretraživanjem

Generiši i testiraj

- Metoda koja se zasniva na prvom navedenom principu. Obuhvata heurističko pretraživanje i koristi se više u oblasti veštačke inteligencije, nego u ekspertskim sistemima, ali se i u ovim sistemima ponekad koristi ovaj princip. Zasniva na:
  - Generatoru (potpuno kandidat rešenje)
  - Procenjivaču (upoređuje predloženo rešenje sa zahtevanim stanjem, sve dok se ne dođe do rešenja)
- Generator bi trebalo da bude potpun, da generiše sva moguća rešenja, i nereduntantan, rešenje se samo jednom generiše. Međutim ovakvo rešenje je samo strukturirano iscrpno pretraživanje.

Generiši i testiraj

- U nekim slučajevima, moguće je naći generator koji će moći da prepozna da čitava klasa predloženih rešenja ne zadovoljava zadato eliminaciono ograničenje - značajno smanjiti prostor pretraživanja. Ovaj proces se naziva hijerarhijsko heneriši i testiraj (HGT).
- **Primer provalnika:** šifra brave: 00-00-00 – ukupno 106 različitih kombinacija. Ako je moguće isprobati tri šifre problema za jedan minut i da se u proseku prođe kroz polovinu kombinacija da bi se uspelo, potrebno je 16 nedelja, ako se radi 24h dnevno.
- Ako provalnik zna da su svi brojevi u kombinaciji prosti, onda ima samo 253 = 15625 brojeva, te bi mu trebalo 2 dana, umesto 16 nedelja.

Generiši i testiraj

- **HGT se često koristi u interpretaciji podataka, odnosno da se pronađe prihvatljiva intrpretacija iz velikog obima podataka.**
- **Posebno efikasno kada je skup prihvatljivih rešenja mali podskup svih rmogućnosti.**
- **Primer spektograma: Određivanje uređenosti atomske strukture u nekom molekulu m na osnovu masenog spektograma razbijanja molekula u naelektrisane čestice. Svrha sistema DENDRAL je da na osnovu hemijske formule, na primer C8H16O, i na osnovu masenog spektograma odredi strukturu, t.j. kako su atomi uređeni u hemijskoj strukturi.**

Transformacija prostora pretraživanja

- **Problem velikog prostora pretraživanja se rešava primenom apstrakcije i problema dekompozicije.**
- **Apstrakcija je proces generalizacije predstave najvažnijih elemenata problema. Ovim se dozvoljava rešavaču problema da ignoriše manje važne detalje i da se usredsredi na glavne odluke, odnosno elemente.**
- **Problem dekompozicije je proces razlaganja složenog problema u skup potproblema čiji je stepen složenosti manji. Hijerahijskim pristupom, moguće je definisati nekoliko nivoa apstrakcije, svaki sa rastućim nivoom detalja.**
- **U najvećem broju slučajeva, potproblemi koji rezultuju iz dekompozicije imaće međusobnu interakciju.**

Suočavanje sa interakcijama

- **Nekada je moguće definisati fiksni redosled izvršavanja obrade potproblema kojim se može kontrolisati interakcija između potproblema.**
- **Syscon (1986) je ekspertski sistem koji se koristi još od 1984.god. radi konfigurisanja velikih računara. Zadatak ovog ekspertskog sistema je da da specifikaciju i konfiguraciju komponenti na niskom nivou koje se zahtevaju da bi se proizvodio sistem koji je u početku opisan kao skup glavnih komponenti. Na primer, dva procesora, U/I kontroler, dve memorijske jedinice, itd.**
- **Postoji 479 različitih tipova komponenti u ciljnom sistemu koje variraju po složenosti od prostog kabla do napajanja. Proces konfiguracije zahteva oko jedne nedelje rada, kada ga radi čovek, i zahteva od 500-800 različitih odluka.**

Fiksni redosled potproblema

- **Iako ima na hiljade dešavanja pojedinačnih komponenti, moguće je posmatrati sistem apstraktno, kao kompoziciju od ne više od 24 različite glavne funkcionalne jedinice. Na osnovu date apstrakcije, ukupan prostor se može podeliti u skup potprostora koji odgovaraju glavnim jedinicama.**
- **Da bi se razmotrio problem interakcije potproblema, oni se planiraju u specifičnom, unapred određenom poretku. Akcije koje se preduzimaju u obradi prvih potproblema ograničavaju mogućnosti kasnije obrađivanih potproblema, i stoga proces postaje jako deterministički, kako proces obrade potproblema odmiče. Ovakav pristup je moguć kod Syscon sistema samo zbog toga što je moguće definisati unapred skup podzadataka i redosled njihovog rešavanja.**
- **Međutim, ovo nije uvek moguće, te je neophodno povećati snagu rezonovanja sistema koja uključuje mogućnosti dinamičkog formiranja skupa potproblema za vreme i u toku obrade problema.**

Planiranje

- **Planiranje je proces odlučivanja o akcijama (redosledu akcija) pre njihovog izvršavanja. Sistemi planiranja se zasnivaju na simuliranju akcija na bazi modela domena. Po simulaciji svake akcije njen efekat na domen se posmatra. Akcije koje se ne mogu korigovati u stvarnom okruženju se odbacuju iz simulacionog modela. Planiranje povećava efikasnost u domenima u kojima se greške mogu ignorisati i korigovati, ali je neefikasno tamo gde su greške nepromenljive.**
- **Sistemi planiranja se zasnivaju na dekompoziciji, da bi se rešio jedan problem, a zatim drugi. Međutim, teškoće se pojavljuju zbog interakcije između problema.**
- **Zato se planeri oslanjaju na dinamičko rešavanje interakcija između potproblema. Polazi se od toga da ti konflikti nisu česti. U takvim slučajevima je razumno pretpostaviti u početku da se problem može potpuno dekompozivati i tek zatim primeniti specijalnu obradu da bi se detektovale i rešile interakcije i to onda kada se dese.**

Planiranje

- **Sistema za planiranje – Strips, koristi ovaj prilaz. Zasniva se pretraživanju na bazi sukcesivnih aproksimacija, uz korišćenje ciljnog steka. Ciljni stek sadrži ciljeve i operatore kojima treba da se ostvare ti ciljevi. Pored ciljnog steka planeri tipa Strips koriste predikatsku logiku za predstavljanje tekućeg stanja sistema, i skup operatora koji se koriste za modifikaciju stanja.**
- **Primena operatora najčešće menja samo neke od objekata (elemenata) koji čine opis stanja sistema. Ako operator čita sve pre i posle uslove, nepromenjeni elementi će biti nepotrebno ponavljani kod svake operacije. Strips rešava ovaj problem na bazi pretpostavke da se svi elementi stanja smatraju nepromenjenim, izuzev onih koji su eksplicitno naznačeni kao promenljivi.**
- **Svakom operatoru je pridružena lista preduslova koji opisuju elemente koji moraju da budu u opisu stanja da bi se operator mogao primeniti, lista dodavanja koja opisuje elemente koji se dodaju u opis stanja po primeni operatora, i lista brisanja koja navodi elemente koje treba ukloniti.**

STRIPS

- **STRIPS sistem za rešavanje problema zasnovan na predikatskoj logici:**
- **Osnovna namena STRIPS-a nije samo dokazivanje da je neki stav moguć, već i generisanje niza operatora čijom se primenom dolazi do željenog stava.**
- **Umesto da koristi situacije kao promenljive koje opisuju koji su iskazi istiniti u određenom stanju, STRIPS održava model sveta, skup izraza koji zajedno kazuju kakva je trenutna situacija.**
- **Procedure (operatori) mogu se aktivirati ako su ispunjeni određeni preduslovi; nakon aktiviranja, određeni stavovi postaju neistiniti I uklanjaju se iz modela sveta, dok neki drugi postaju istiniti I moraju se dodati modelu sveta.**
- **STRIPS može da posluži kao heuristički rešavač problema.**

STRIPS - algoritam

1 Formirati opis trenutnog stanja i inicijalizovati ciljni stek, na osnovu opisa početnog, odn. Ciljnog stanja.

2 Pretraživanjem po dubini ispitati vrh ciljnog steka, sve dok se ne uspe, ili dok se ne iscrpu sve alternative (tačke grananja):

2a. Ako vrh steka odgovara trenutnom stanju, izvršiti potrebne unifikacije ukloniti (pop) vrh steka.

2b. Ako je na vrhu steka cilj sa jednom komponentom:

2b1 pronaći operator koji sadrži takvu komponentu (stav) u svom DODAJ delu (tačka grananja!),

2b2 izvršiti unifikaciju stavova iz DODAJ dela I stava sa vrha ciljnog steka (tačka grananja),

2b3 staviti operator na vrh steka (push),

2b4 staviti preduslov toga operatora na vrh steka (push)

2c. Ako je na vrhu steka operator, ukloniti iz modela sveta stavove koji su navedeni u UKLONI delu, pa zatim dodati modelu sveta stavove navedene u DODAJ delu, pa zatim dodati modelu sveta stavove navedene u DODAJ delu (definicije operatora). Operator dopisati u plan operacija.

2d. Ako je na vrhu steka složeni cilj, staviti svaku od njegovih komponenata na vrh steka. Redosled stavljanja nije dat (tačka grananja), mada očigledno od toga redosleda može da zavisi dalji tok događaja.

2e. Ako je ciljni stek prazan (što znači da model sveta odgovara ciljnom stanju), objavi USPEH.

STRIPS - primer

- **Posmatra se situacija u kojoj su date neke kocke (cigle) na stolu, a potrebno je generisati niz operacija premeštanja kocki (uz pomoć mehaničke hvataljke), kako bi se dobio neki drugi zadati raspored.**
- **U polaznom stanju model sveta je opisan sledećim stavovima:**

Na (B,A) **NaVrhu(B)** **NaStolu(A)**

NaStolu(C) **NaVrhu(C)**

NaStolu(D) **NaVrhu(D)**

RukaPrazna

- **Ciljno stanje**

STRIPS - primer

- **Mehanička hvataljka (ruka), operatori:**
- **UZMI(x)**
  PREDUSLOV: ***RukaPraznaNaVrhu(x) NaStolu(x)***
  UKLONI: ***RukaPrazna; NaVrhu(x); NaStolu(x)***
  DODAJ: ***URuci(x)***
- **SPUSTI(y)**
  PREDUSLOV: ***URuci(y)***
  UKLONI: ***URuci(y)***
  DODAJ: ***RukaPrazna; NaVrhu(y); NaStolu(y)***
- **SKINI(u,z)**
  PREDUSLOV: ***RukaPraznaNaVrhu(u) Na(u,z)***
  UKLONI: ***RukaPrazna; NaVrhu(u); Na(u,z)***
  DODAJ: ***URuci(u), NaVrhu(z)***
- **STAVI(v,w)**
  PREDUSLOV: ***URuci(y) NaVrhu(w)***
  UKLONI: ***URuci(y); NaVrhu(w)***
  DODAJ: ***RukaPrazna; NaVrhu(v); Na(v,w)***

STRIPS - primer

- **Inicijalizuje se ciljni stek opisom ciljnog stanja:**
- ***Na(C,A) NaVrhu(C) NaStolu(A) Na(B,D) NaVrhu(B) NaStolu(D)***
- **Korak 2. Na stek se stavljaju komponente složenog cilja :**

$Na(C,A) \wedge NaVrhu(C) \wedge NaStolu(A) \wedge Na(B,D) \wedge NaVrhu(B) \wedge NaStolu(D)$

Na(C,A)
NaVrhu(C)
NaStolu(A)
Na(B,D)
NaVrhu(B)
NaStolu(D)

STRIPS - primer

Na(C,A) ∧ NaVrhu(C) ∧ NaStolu(A) ∧ Na(B,D) ∧ NaVrhu(B) ∧ NaStolu(D)
Na(C,A)
NaVrhu(C)
NaStolu(A)
Na(B,D)
STAVI(B,D)
URuci(B) ∧ NaVrhu(D)

Na(C,A) ∧ NaVrhu(C) ∧ NaStolu(A) ∧ Na(B,D) ∧ NaVrhu(B) ∧ NaStolu(
Na(C,A)
NaVrhu(C)
NaStolu(A)
Na(B,D)
STAVI(B,D)
URuci(B)
SKINI(B,A)
RukaPrazna ∧ NaVrhu(B) ∧ Na(B,A)

STRIPS - primer

- *Prema koraku 2c, operator se skida sa steka i upisuje u plan operacija:*
- *SKINI(B,A)*
- *i vrši se modifikovanje trenutnog stanja sveta: uklanjaju se stavovi koje operator SKINI ima u svom UKLONI delu, a dodaju se stavovi iz njegovog DODAJ dela. Posle ovoga stanje sveta izgleda ovako:*

- *i opisano je sledećim stavovima*

NaStolu(A)	*NaVrhu(A)*
NaStolu(C)	*NaVrhu(C)*
NaStolu(D)	*NaVrhu(D)*
URuci(B)	

STRIPS - primer

Dok ciljni stek izgleda ovako:

Na(C,A) $\wedge$ *NaVrhu(C)* $\wedge$ *NaStolu(A)* $\wedge$ *Na(B,D)* $\wedge$ *NaVrhu(B)* $\wedge$ *NaStolu(D)*

Na(C,A)

NaVrhu(C)

NaStolu(A)

Na(B,D)

STAVI(B,D)

URuci(B)

STRIPS - primer

- Prema koraku 2c, operator se skida sa steka dopisuje u plan operacija, koji sada izgleda ovako:

  SKINI(B,A)

  STAVI(B,D)

- Iz stanja sveta uklanjaju se stavovi iz UKLONI dela definicije operatora STAVI, a dodaju se stavovi iz DODAJ del ate definicije. Svet sada izgleda kao na slici:
- I opisan je sledećim stavovima:

  NaStolu(A) *NaVrhu(A)*

  NaStolu(C) *NaVrhu(C)*

  Na(B,D) *NaVrhu(B)* *NaStolu(D)*

  RukaPrazna

STRIPS - primer

Dok ciljni stek, posle uklanjanja stavova koji u ovom stanju važe izgleda ovako:

Na(C,A) $\wedge$ *NaVrhu(C)* $\wedge$ *NaStolu(A)* $\wedge$ *Na(B,D)* $\wedge$ *NaVrhu(B)* $\wedge$ *NaStolu(D)*

Na(C,A)

Sada tražimo operator koji ima stav tipa Na(C,A) u svom DODAJ delu, a to je slučaj jedino sa operatoraom STAVI. Potrebno je izvršiti unifikacije v$\Leftrightarrow$C, w$\Leftrightarrow$A, posle čega se na stek stavljaju operator STAVI(C,A), a zatim i njegovi preduslovi. Kako je preduslov složen, na stek se stavljaju pojedinačno i njegove komponente, posle čega stek izgleda ovako:

Na(C,A) $\wedge$ *NaVrhu(C)* $\wedge$ *NaStolu(A)* $\wedge$ *Na(B,D)* $\wedge$ *NaVrhu(B)* $\wedge$ *NaStolu(D)*

Na(C,A)

STAVI(C,A)

URuci(C) $\wedge$ *NaVrhu(A)*

URuci(C)

NaVrhu(A)

STRIPS - primer

Na(C,A) ∧ NaVrhu(C) ∧ NaStolu(A) ∧ Na(B,D) ∧ NaVrhu(B) ∧ NaStolu(D)

Na(C,A)

STAVI(C,A)

URuci(C) ∧ NaVrhu(A)

URuci(C)

UZMI(C)

RukaPrazna ∧ NaVrhu(A) ∧ NaStolu(A)

RukaPrazna

NaVrhu(A)

NaVrhu(A)

STRIPS - primer

Plan operacija sada izgleda ovako:

1. SKINI(B,A)
2. STAVI(B,D)
3. UZMI(C)

Dok ciljni stek ima sledeći izgled:

$Na(C,A) \wedge NaVrhu(C) \wedge NaStolu(A) \wedge Na(B,D) \wedge NaVrhu(B) \wedge NaStolu(D)$

$Na(C,A)$

STAVI(C,A)

$URuci(C) \wedge NaVrhu(A)$

$URuci(C)$

STRIPS - primer

- Stanje sveta prikazano je na slici
- Što znači da važe sledeći stavovi:

NaStolu(A) *NaVrhu(A)*

Na(B,D) *NaVrhu(B)*

URuci(C) *NaStolu(D)*

STRIPS - primer

- *Postignuto je ciljno stanje. Plan operacija:*

  SKINI(B,A)

  STAVI(B,D)

  UZMI(C)

  STAVI(C,A)

STRIPS - primer

- Teme za razmišljanje:
  - način izbora operatora, ako ih ima više koji se mogu primeniti
  - redosled stavljanja komponenata složenog cilja na stek
  - način detektovanja pogrešnog izbora operatora/neispravnog redosleda na steku

Princip najmanjeg angažovanja i propagacije ograničenja

- Akcije koje se zahtevaju da se dostigne cilj se započinju i završavaju pre nego što počne da se radi na sledećem cilju, odnosno akciji.
- To omogućava koordinaciju aktivnosti odlučivanja od raspoloživosti informacija izbegava se mogućnost pogrešnih odluka. U mnogim slučajevima je poželjno da se izvede nelinerani plan po kome se akcije vezane za neki cilj mogu prekinuti i ponovo nastaviti po ispunjavanju drugog cilja.
- Jedan prilaz generisanju takvih planova je princip najmanjeg angažovanja. Odluke se odlažu do poslednjeg momenta kada je na raspolaganju maksimalna količina informacija.

Klasifikacioni model

- To je strategija rešavanja problema kojom se strukturiraju pravila, posebno za dijagnostičke i interpretativne zadatke. CM organizuje rezonovanje koje se zasniva na klasifikaciji: selektuju se zaključci iz unapred specificirane liste mogućih zaključaka.
- CM je realizovana kao modifikovani produkcioni sistem sa bazom znanja, kontrolnim mehanizmom i radnom memorijom.
- Baza znanja je:
  - lista svih mogućih opažanja (ono što se početno gubilo i nalaženja koja prozilaze iz i testova ili eksperimenata koja se koriste za sakupljanje podataka)
  - lista mogućih zaključaka (uključuje posredne i konačne zaključke. U većini slučajeva proces rezonovanja se pomera kroz nekoliko nivoa posrednih zaključaka pre nego što se stigne do konačnih)
  - skup pravila koja povezuju opžanja sa zaključcima

Klasifikacioni model

Kontrolni mehanizam:

- uređuje skup prukupljenih podataka, odnosno opažanja

Radna memorija:

- sadrži početna opažanja, podatke do kojih se došlo i zaključke koji su izvedeni i do kojih se stiglo. Radna memorija CM je prostija nego kod produkcionih sistema, jer se radi o propozicionalnoj logici i iskazima koji su tačni ili nisu.
- Proces rezonovanja je jednostavan. Zasniva se na predstavi implikacija putem pravila i primena modus ponens na osnovu prikupljanja podataka. Proces počinje početnim opažanjem i primenjuju se pravila da bi se došlo do novih zaključaka ili posrednih hipoteza. Zatim se nastavlja kroz nekoliko nivoa posredne analize dok se ne dođe do klasičnog zaključka.

Klasifikacioni model

Koriste se dva nivoa pravila:

- podatak/zaključak: ova pravila se koriste da se navedu zaključci koji su inicirani podacima i da se upravlja procesom rezonovanja na napred opisani način. Primer: ***if*** *element koji se greje u rerni postaje crven* ***then*** *zaključiti da je element vreo*.
- Zaključak/podatak: ovaj tip pravila opisuje fakt, događaj, podatak, koji je morao da se desi akopostoje dati uslovi. Ova pravila se koriste da opišu podatak koji treba tražiti da potvrdi hipotezu. Primer: ***if*** *svestlost nestala zbog gubitka napajanja u trafostanici,* ***then*** *susedna svetla takođe mora da su ugašena.*

Klasifikacioni model

Unutar ovih opštih klasa postoje specifični tipovi pravila:

- Nalaženje/nalaženje: ova vrsta pravila opisuje činjenicu da se jedno nalaženje može izvesti iz drugog nalaženja. Na primer pravilo kaže da ako se utvrdi da je akumulator prazan, tada se može utvrditi da je nalaženje da svetla na kolima ne rade takođe istinito.
- Nalaženje/hipoteza: ova vrsta pravila se koriste da se generišu posredne hipoteze (zaključci) iz opažanja ili nalaženja. Na primer *ako je osigurač iskočio ili prgoreo, tada zaključiti da postoji električni problem*.
- Hipoteza/hipoteza: ova vrsta pravila se koriste da se pomeri od jedne hipoteze do druge ili konačnog zaključka. Primer: ***if*** *iznenadno nestane napajanje i sva svetla u susedstvu su takođe nestala* ***then*** *zaključiti da je trafostanica izgubila napajanje*

Model školske table

- To je arhitektura ekspertskog sistema koja se koristi da se strukturira rezonovanje u kompleksnom domenu. Prostor rešenja se deli u hijerarhiju parcijalnih rešenja, svaki na različitim nivoima apstrakcije. Na primer kod sistema koji interpretira izgovorene reši parcijalni nivoi rešenja su: glasovi, segmenti, slogovi, reči, sekvence reči, rečenice.
- Ekspertski sistem za dijagnozu problema u velikim računarskim sistemima: hardverske greške pojedinih jedinica, softverske greške pojedinih jedinica, podsistemske greške, međusistemske greške.
- Znanje u ovom modelu je razdvojeno u posebne izvore znanja koje mogu da opažaju i modifikuju sadržaj modela ali ne mogu međusobno komunicirati.

Model školske table

- Obrada ovog modela se zasniva na komunikaciji nezavisno kooperirajućih eksperata. Kada su suočeni sa kompleksnim problemima, eksperti rade zajedno doprinoseći svaki parcijalnom rešenju koje odgovara njegovoj oblasti. Na primer u dijagnostici, hardverski, softverski i sistemski ekspert su zajedno uključeni u rešavanje složenih računarskih problema, jer se bilo koji dati problem može manifestovati u različitim oblicima.
- Svaki ekspert osmatra tekući sadržaj modela i pokušava da razvije novo parcijalno rešenje zasnovano na sopstvenom poznavanju tekućeg stanja. Tako u pomenutom sistemu za interpretiranje izgovorenih reči kada se skup slogova pojavi u modelu ekspert za nivo reči pokušava da formira reči iz slogova.