Mesterséges intelligencia 1

Elõadó: Dr. Várterész Magdolna

2.2. Állapottér-reprezentált problémák megoldását keresõ módszerek

Komponensek:

-adatbázis

Az állapottérgráf tárban tárolt része. Megoldáskeresésnél nem lesz az egész gráf a tárban, csupán valamilyen, a keresés szempontjából fontos része.

-mûveletek

Segítségükkel módosítjuk az adatbázist. Ilyen mûveletek például:

a) állapottér-reprezentációs operátorokból származtatott mûveletek

b) technikai mûveletek (pl. visszalépés)

-vezérlõ

A vezérlõ irányítja a keresést. Megmondja, hogy a megoldáskeresés során az adatbázis mely részén mikor, melyik mûvelet hajtódjon végre. Operátorból származtatott mûvelet választása elõtt vizsgálja az operátor-alkalmazási elõfeltételeket. Figyeli a terminálási feltételek segítségével, hogy befejezõdhet-e a keresés (sikeresen, avagy sikertelenül). Ha a terminálási feltételek nem teljesülnek, akkor tovább kell keresni.

Ha ezek nem teljesülnek: még nem találtuk meg a megoldást. Õ fogja figyelni is.

Osztályozásuk:

I. nem-módosítható megoldáskeresõk: egy állapotváltozást vissza lehet-e vonni, vagy sem. Ha utólag nem tudjuk magunkat módosítani: nem-módosítható megoldáskeresés (ez a hagyományos programozásra jellemzõ). Mesterséges intelligenciában inkább módosítható megoldáskeresõket alkalmaznak.

II. módosítható megoldáskeresõk: a vezérlõ által kiválasztott mûvelet hatása visszavonható

a) visszalépéses (backtracking)

b) keresõgráffal

III. irányítatlan (szisztematikus): a vezérlõ mi alapján választ:

- véletlenszerûen
- valamilyen általános szisztéma alapján (pl. gráfban fentrõl le, stb.)

IV. heurisztikus: Heurisztikus keresésnél a generálás irányításánál helyet hagyunk a tárgyköri ismereteknek is, azokat is felhasználjuk.

Jelentõsége: megpróbáljuk a keresést a reprezentációs

gráfban ott folytatni, ahol a megoldást reméljük (valamilyen becslés alapján) → csökken a reprezentációs gráf tárban tárolt részének mérete

Keresés iránya:

1. elõrehaladó(adatvezérelt): kezdõállapotból célállapotba
2. visszafelé haladó (célvezérelt): visszafelé haladva rekonstruálunk
3. kevert: mindkét irányból elindul, s valahol találkozik

Linkek:

- nem-módosítható megoldáskeresõk
- módosítható megoldáskeresõk
  1. visszalépéses (backtracking)
  2. keresõgráffal

Vissza a lap tetejére

Mesterséges intelligencia 1

Elõadó: Dr. Várterész Magdolna

2.2.1. Nem módosítható megoldáskeresõk

Ezen megoldáskeresõknek kisebb a jelentõségük, ritkábban is használják õket. Elõnyük viszont az, hogy egyszerûbbek. Csak olyan feladat megoldásánál alkalmazzuk, ahol nem a megoldás a lényeg (azaz a kezdõállapotból a célállapotba vezetõ operátorsorozat), hanem csupán az, hogy eldöntsük: a feladatnak egyáltalán létezik-e megoldása, vagy egy célfeltételnek eleget tevõ állapot elõállítása.

Komponensek:

adatbázis: aktuális állapot

mûveletek: állapottér-reprezentációs operátorok

Az adatbázisra egy mûvelet alkalmazható: ha az aktuális állapotban teljesülnek a mûveletet meghatározó operátor alkalmazási elõfeltételei. A mûvelet hatása az adatbázisra: az aktuális állapotban alkalmazzuk a mûveletet meghatározó operátort, és az elõállt új állapot lesz az új aktuális állapot.

vezérlõ:

1. aktuális állapot ← kezdõállapot

   (inicializáljuk az adatbázist. Az aktuális állapot legyen elõször a kezdõállapot)

   Célállapotot kell elõállítani, nem egy operátorsorozatot (hiszen ez már plusz információ lenne).

2. tesztelés. Az aktuális állapot egyenlõ célállapot-e?

   Ha igen: sikeresen befejezhetjük a keresést,
   elõállítottunk egy célállapotot

   Ha nem: erre az aktuális állapotra van-e alkalmazható mûvelet?

   a) ha van: a vezérlõ választ egyet, és alkalmazza.
   Vissza a 2. pontra (tesztelés).

   b) ha nincs: be kell fejezni a munkát, sikertelen a keresés.
   Lehet, hogy nincs megoldás, vagy a megoldáskeresõ rossz irányba ment; de errõl nem tudunk többet mondani!

A szükséges eljárások:

`procedure alk (var all: alltip; o: op);`	mûvelet alkalmazása az adatbázison
`procedure valaszt (all: alltip; var o: op; var v: boolean);`	mûvelet választása
`procedure sikertelen;`	felhasználó tájékoztatása a keresés sikertelen befejezésérõl
`procedure sikeres (a: alltip);`	megoldás visszaadása

Ezek alapján a vezérlõ:

```
begin
    aktall := k; van := true;
    while van and (not celfelt(aktall)) do
    begin
        valaszt(aktall, operator, van);
        if van then alk(aktall,operator);
    end;
    if van then sikeres(aktall) else sikertelen;
end.
```

Ugyanazon feladat esetén a választás módjában lehet lényeges eltérés:

a) irányítatlanul, szisztematikusan ("próba-hiba" módszer)

Pld.: sorrend, s ezeket végignézzük, hogy melyiket tudjuk alkalmazni. A feladattal kapcsolatos tudásunk nincs benne megfogalmazva. Véletlenszerûen választunk: ez a próba-hiba módszer.

Ezt könnyû implementálni, kicsi az adatbázis, viszont általános esetben nem garantál semmit. Milyen feladatosztály esetén van reményünk a megoldásra? Csak olyan esetben, ha minden csúcsból elérhetõ valamelyik cél. Kört nem tartalmazhat, ui. nincs rá garancia (végtelenszer ismételheti). Hurok még lehet benne.

b) heurisztikusan ("hegymászó" módszer)

Akkor beszélünk heurisztikus megoldáskeresésrõl, ha megpróbáljuk felhasználni a tudásunkat, tapasztalatunkat. A heurisztika valamilyen mennyiségi, minõségi heurisztika lesz.

A heurisztikától függ majd, hogy a megoldást megtaláljuk-e vagy sem. Jó heurisztika esetén lesz majd reményünk a megoldás megtalálására.

h(n) a csúcs hány élnyire van a legközelebbi terminálishoz (azaz hány operátor alkalmazására van szükség). Ha ezt tudnánk, akkor ez egy felhasználható tudás lenne. Ezt próbáljuk meg becsülni.

h(aktall)	$h(o_1(aktall))$	alkalmazzuk az összes lehetséges operátort,
	$h(o_2(aktall))$	s utána megnézzük, hogy közelebb kerültünk-e
	...	valamilyen célhoz (vagy legalábbis nem távolabb).
	$h(o_n(aktall))$	Azt fogjuk alkalmazni, amelyik a legközelebbi célhoz visz közelebb.

Ez az ún. hegymászó-módszer. "A csúcshoz közelítünk." Távolodni nem szabad a céltól.

TOC

Vissza a lap tetejére

Mesterséges intelligencia 1

Elõadó: Dr. Várterész Magdolna

2.2.2. Módosítható megoldáskeresõk

Információ kellene tárolni az információkeresés múltjáról. Ha felismerjük, hogy rossz irányba haladunk, akkor visszalépünk.

Módosítható megoldáskeresõk osztályozása:
1. visszalépéses (backtrack)
2. keresõgráffal

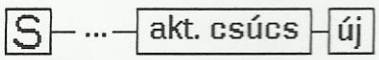
2.2.2.1. Visszalépéses (backtrack)

Adatbázis: a keresõgráf egy része kerül ide. `start`-ból induló, aktuális csúcsba vezetõ aktuális út. Ez az az út amibõl véljük, hogy elérjük valamelyik terminális csúcsot. A múltból mi van nyilvántartva: a `start`-ból milyen módon jutottunk el az aktuális csúcsba.

Mûveletek: operátorok + visszalépés

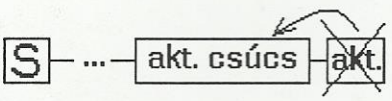
operátorok: állapottérreprezentációs mûveletek
visszalépés: technikai mûvelet

Operátor: alkalmazzuk az utolsó csúcson, ezzel új állapot keletkezik. Ezt felfûzzük az aktuális útra (annak hossza 1 élnyivel megnõ). Ez lesz az új aktuális csúcs.



Az operátor alkalmazása tehát növeli az utat!

Visszalépés: töröljük az aktuális csúcsot, és az aktuális csúcs a megelõzõ csúcs lesz. Vagyis az út rövidebb lesz.



Vezérlõ: eldönti, hogy az adatbázis melyik részén mikor, melyik mûveletet kell végrehajtani (az aktuális csúcsra fog alkalmazni egy mûveletet).
Mit figyeljen?

- operátor vagy visszalépés? Ezt kell eldöntenie. Ha operátor: melyiket alkalmazza?
- terminálási feltételek teljesülése. Elõállt-e a megoldás? Figyeli az akt. csúcsot. Ha az teljesíti a

célfeltétel(eke)t, akkor kész vagyunk. Az adatbzis tehát egy megoldás.

Ha nemterminális a csúcs, akkor ezen az úton remény van-e még a megoldás megtalálására vagy sem? Vagy: érdemes-e még folytatni egyáltalán a keresést?

Hogy fog a vezérlõ megoldást keresni?

a) "alap" backtrack

1. inicializálás. Az út egy csúcsból áll: S (start csúcs)
   Van adatbázis, aktuális út, aktuális csúcs

2. tesztelés. Az aktuális csúcs terminális-e?
   igen: kész vagyunk
   nem: tovább tudunk-e lépni? Van-e alkalmazható operátor?
   igen, van: választunk, alkalmazzuk. Új aktuális csúcs, ismét tesztelés (2. pont)
   nincs: másik mûvelet: visszalépés.
   (vagyis ha az akt. csúcsban nincs alkalmazható operátor)

Ha az aktuális csúcsban nincs alkalmazható operátor, akkor meg kell próbálni a megoldást valamilyen másik irányban keresni. A reprezentációs gráfban valamilyen másik irányba kellene lépni. Az adatbázisnak viszont ezt tudnia kell, minden csúcsban nyilván kell tertani, hogy abból a csúcsból mely alkalmazható operátorokat alkalmaztuk, vagy nem alkalmaztuk (ez egy plusz információ).

A visszalépés során elõállt új aktuális csúcsban egy még nem alkalmazott alkalmazható operátort kell alkalmazni.

Van ilyen?
igen: alkalmazzuk, folytatjuk a keresést
nem: visszalépés a szülõre, s a reprezentációs gráfban egy másik irányba kell próbálkozni. Vagyis: aktuális csúcsban már minden alkalmazható operátort kipróbáltunk, sikertelenül F visszalépés.

Terminálási feltételek:
1. az aktuális csúcs célcsúcs-e?
2. `start` csúcsban teljesül a visszalépési feltételünk
   (vagyis már nincs olyan operátor, amit ne alkalmaztunk volna már sikertelenül)
   Ekkor nincs megoldás!

Ha a reprezentációs gráfunk olyan véges gráf, hogy nem tartalmaz köröket, a visszalépéses megoldáskeresõ - ha van megoldás - garantáltan elõállít egy lehetséges megoldást. Ha nincs megoldás, akkor azt fel lehet ismerni.

Eltérés hol lehet? Az operátorok választásában. Ez történhet:

- szisztematikusan
- heurisztikusan

Heurisztika: tudásunk alapján megbecsülhetjük, hogy a cél milyen távol van. Ezért választhatunk olyat, ami közelebb visz (hasonló a hegymászó módszerhez).

Mik a problémák? Olyan reprezentációs gráfok esetén garantálja a megoldást, amelyben nincsenek körök. Ekkor ui. nincs garantálva a megoldás, végtelen ciklus alakulhat ki. (Végtelen sokszor ugyanazt választjuk ki).
Ötlet: ne engedjük, hogy egy körön végtelen sokszor végigmenjen, vagy ne is engedjük bezárni ezt a kört:

b) kör kiküszöbölése

lesz egy újabb visszalépési feltétel: az aktuális csúcs szerepelt-e már az aktuális úton

Ha igen: rögtön visszalépés (így nem zárjuk be a kört).
Azaz körök nélkül próbáljuk meg megtalálni a megoldást.
(Ha van megoldás, akkor van körmentes megoldás is).

c) úthossz-korlát

Egy másik megoldás az, hogy úthossz-korlátot vezetünk be: [ ]l (megoldás)
Fává egyenesítünk, végtelen fagráfot állítunk elõ. Nem engedjük, hogy az aktuális út hossza meghaladja az úthosszkorlátot.

aktuális út hossza ≥ úthossz-korlát (ha ez teljesül [ ] visszalépés)

Viszont ha túl rövidre választjuk az úthossz-korlátot (túl alacsonyan vágjuk el a gráfot) akkor nem találunk megoldást.

Ha a `start` csúcsban áll elõ a visszalépési feltétel, akkor:

1. nincs megoldás
2. túl rövidre választottuk az úthossz-korlátot

Arról, hogy melyik, nem ad információt!

Ez utóbbi (úthossz-korlátos) tartalmazhat kört a megoldásban. Ha körmenteset akarunk, akkor onnan nekünk kell kivágni a kört.

Nem feltétlenül véges fa-gráf esetén, ha garantáltan van az úthossz-korlátnál rövidebb megoldás, akkor azt megtalálja.

d) optimális megoldás keresése

A backtrack alkalmas-e optimális megoldás keresésére? Azaz van költség, s a legkisebb költségû megoldást szeretnénk elõállítani. Ez lesz az ág és korlát algoritmus.

van egy induló költségkorlát [ ]k (megoldás) (felsõ becslés)

Ennél a költségkorlátnál nem költségesebb megoldást keresünk. Csak addig megyünk a reprezentációs gráfban a keresés során, míg a költségkorlátot el nem érjük.

Ha találunk egy megoldást, akkor ez lesz az új költségkorlát! (Ez ugye $\leq$ induló költségkorlát)
Folytatjuk a megoldáskeresést. Ezt a megoldást elraktározzuk, s úgy tekintjük, mintha nem lett volna megoldás. Ha van újabb megoldás, akkor annak költsége már csak $\leq$ lehet, mint ez. Ismét csere, eltárolás, stb.
Így a reprezentációs gráf elég nagy részét nem kell bejárni, egyre kisebb utakat járunk be. Addig folytatjuk, míg a `start` csúcsból el tudunk valamerre indulni.

Mit garantál ez?

- elõállítja az optimális megoldást
- ha nem talált megoldást, akkor vagy a költségkorlát volt kicsi (az induló), vagy a feladatnak nincs megoldása.

CSAK körmentes gráfok esetén mûködik!

Vissza a lap tetejére